NASA Contractor Report 179434

Model Reduction Methods for Control Design

K.R. Dunipace

Contract NCC2-289

August 1988

NASA
National Aeronautics and
Space Administration

NASA Contractor Report 179434

Model Reduction Methods for Control Design

K.R. Dunipace
Engineering Division, Indiana University-Purdue University at Indianapolis, Lafayette, Indiana

Prepared for
Ames Research Center
Dryden Flight Research Facility
Edwards, California
under Contract NCC2-289

1988

National Aeronautics and
Space Administration
Ames Research Center
Dryden Flight Research Facility
Edwards, California 93523-5000

MODEL REDUCTION METHODS FOR CONTROL DESIGN

TABLE OF CONTENTS

"MODEL REDUCTION METHODS FOR CONTROL DESIGN"

(Final Report for NASA-Ames Cooperative Agreement No. NCC 2-289)

K. R. Dunipace

November 7, 1987

ABSTRACT

Analytical methods lead to high-order models for complicated systems, while low-order models are more desirable for study and design. A computer program has been developed to extract low-order models from high-order models of linear systems. Several system models have been developed to demonstrate the feasibility of the methods and to illustrate specific difficulties in model reduction. A User's Guide has been prepared and is attached.

PROJECT DESCRIPTION

A. INTRODUCTION:

Linear dynamic models of aircraft and other complex engineering systems are often represented by high-order linear differential equations. Such models are usually derived through complicated analytical procedures and involve predicted values of the many system parameters. The complexity of such models provides a serious handicap to understanding system behavior. Similarly, the cumulative effects of uncertainties in predicting parameter values frequently lead to unreliable accuracy and misleading characteristics in the high order models. In addition, low-order models are important in the design of robust controls.

The objective of this project was to develop usable and reliable procedures for extracting and evaluating low-order models which retain the essential dynamics of the real system and provide desirable insight into performance characteristics needed in design studies. Such procedures would have been of value in the design of the Manuever Autopilot used in the Hi-Mat aircraft. They could be of similar value in implementing control laws for the X-29 and similar advanced aircraft.

B. STATUS:

An interactive computer program has been developed (in MATRIX$_x$

language) to implement five different methods of model reduction. The program has been demonstrated and the five reduction methods evaluated in reducing the unit-step response of four or more demonstration system models. The two methods directly applicable to multivariable systems were also evaluated for performance in reducing a fourth-order unstable model of the X29, a fifth-order unstable model of an F4, a tenth-order stable model of an F15, and a 45th-order stable model of the X29. An extensive computer data-base literature search was conducted and the resulting references were reviewed for additional model reduction techniques. No additional methods were developed as a result of this review, though some methods were found which may offer promise (see G. Recommendations for Further Study).

C. DEMONSTRATION SYSTEM MODELS:

Five basic demonstration system models (with as many as three versions of each) have been developed and used to compare and evaluate the performance of model reduction algorithms. Each model was selected to demonstrate a particular aspect of the model reduction problem. Several additional models have been developed to evaluate and demonstrate the application of the multivariable methods to both continuous and discrete models of contemporary aircraft. The demonstration models and their purposes are discussed below. The remaining models are discussed in Appendix B of the User's Guide.

Throughout the following discussion, demonstrations assume that a third-order system is to be reduced to a second-order system.

1. System #1 - ONE NEGLIGIBLE RESIDUE

The first system model, chosen largely for tutorial purposes, has a real eigenvalue with one residue which is much smaller than the rest. The system transfer function and step response are:

$$T(s) = \frac{0.48(S+5)(S+5)}{(s+1)(S+2)(S+6)}$$

$$Y_u(t) = 1 - 0.004e^{-6t} + 0.54e^{-2t} - 1.54\ e^{-t}$$

It is apparent that the response is hardly changed if the mode at -6 is omitted. Thus, this third-order model can be accurately represented by a second-order model and any worthwhile model reduction should perform well with this model.

2. System #2 - ALL EQUAL RESIDUES

In this model each of the eigenvalues contributes equally to the response. Thus, it is not apparent whether there is a good second-order model. If such a model exists, its

discovery will require relocation of the eigenvalues and provide a more severe test of model reduction algorithms. The system transfer function and step response are:

$$T(s) = \frac{2.5(S+0.75)(S+1.5)}{(S+1)(S+1.25)(S+2.25)}$$

$$Y_u(t) = 1 - \epsilon^{-2.25t} - \epsilon^{-1.25t} + \epsilon^{-t}$$

3. System #3 - COMPLEX ROOTS

In this model the smaller residues occur as a complex pair at the complex eigenvalues of the system. Again, it is not apparent that a good second-order model exists. If such a model exists, its discovery will require relocation of complex eigenvalues. This test is different from, and generally more difficult than the test posed by system #2. The system transfer function and step response are:

$$T(s) = \frac{1.11(S+0.5)(S+0.5)}{(s+1)(S+0.5-j0.167)(S+0.5+j0.167)}$$

$$y_u(t) = 1 - \epsilon^{-t} + 0.667\epsilon^{-0.5t}\sin(0.167t)$$

4. System #5 - UNSTABLE REAL ROOT

In this model one of the real roots is in the right-half plane. Thus, the system is unstable and the response to the unstable root grows without bound. Reduction of such a model requires selection of a significant time period so that the algorithm can assess the relative importance of response contributions within that time.

5. System #9 - NON-MINIMUM PHASE

In this model one of the real zeroes is in the right-half plane. The model was to assess the effect of non-minimum phase properties on algorithm performance. Non-minimum phase had no apparent effect on any of the algorithms.

6. Other System Models

A number of other models have been used for special purposes. A second-order unstable model was used in developing a "balancing" algorithm for unstable systems. Unstable third-order models analagous to the existing stable models were developed and incorporated in routine evaluation of algorithms. Additional models were developed to verify and evaluate algorithm performance for multivariable systems. Present models for multivariable systems include an unstable fourth-order model for the X29, a stable fifth-order model

for the F4 longitudinal dynamics (ref. 1, pp. 4-5), an unstable tenth-order model for the F15 (ref. 2, pp. 47-48), and a stable 45th-order model for the X29.

D. REDUCTION METHODS AND RESULTS:

Five reduction methods have been implemented and evaluated. Of these, only the continuous and discrete versions of "Subsystem Elimination" (ref. 3) are directly applicable to multivariable systems. The five methods are discussed below. A table showing comparative performance follows the discussion.

1. Root Selection

Root Selection is the classical method of time-domain model reduction (ref. 4, pg 184). The residues of the step response are observed and the smaller ones discarded until the remaining roots represent the desired model order. The method does not work well for responses with nearly equal residues. It requires that complex roots be retained or discarded in pairs. It is not convenient for multivariable systems. Its principal value is in developing an understanding of model reduction.

2. Residue Retention

Residue Retention (ref. 2, pp. 21ff) represents a modification of the Root Selection method, above, to provide a time-average adjustment to the step response to provide steady-state compensation for the time-varying terms that are discarded. The method offers modest improvement over the Root Selection method when residues are nearly equal. The method displays the other limitations of the Root Selection method. It is not competitive with other methods.

3. Differentiation of Polynomials

Differentiation of Polynomials (ref. 5) is a convenient paper-and-pencil algorithm which permits easy (though possibly tedious) reduction of a transfer function model. Performance of the algorithm consistently provides results which are a reasonable visual approximation of the actual response. The algorithm, however, has never done especially well by the performance criteria used in this study.

4. Subsystem Elimination

Subsystem Elimination (ref. 3, pp. 26ff) is a mathematically complicated algorithm based on Singular Value Decomposition, the Grammian, and the second-order modes of the system. This method is one of several which require that the system be similarity transformed to "equalize" the effects of inputs on states and of states on outputs. The

method has worked well on all system models studied. Original algorithms were developed for discrete-time and for unstable systems. While not automatic like the stable continuous version all have shown the capability of producing reduced models with good performance.

5. Comparative Results

An RMS error criteria has been used to evaluate and compare algorithms. The formula below has been used as a performance criteria for evaluating and comparing algorithms:

$$\text{Error} = \frac{1}{npts}\left(\sum_{k=1}^{npts} \sum_{j=1}^{nc} \left\{ \frac{\left[y_f(j,k) - y_r(j,k) \right]^2}{y_f(j,k)} \right\} \right)^{\frac{1}{2}}$$

npts ≜ No. of points where response is plotted

nc ≜ Number of system output variables

$y_f(j,k)$ ≜ Value of jth output of full system at kth time instant

$y_r(j,k)$ ≜ Value of jth output of reduced system at kth time instant

The resulting performance for the four algorithms and three system models discussed above is shown in figure 1. The Subsystem Elimination method was clearly superior to the other algorithms in each case. Figures 2 through 5 show representative step-responses for the four different algorithms.

E. MODEL REDUCTION PROGRAM:

An interactive computer program, MODRED, (in MATRIX$_x$ language) has been developed implementing the five model reduction algorithms described above. The program is modularized to provide for easy addition of new algorithms or deletion of algorithms which prove to be of little use. Similarly, the structure of the program makes it easy to change the routine output of the program. The character of the MATRIX$_x$ language provides for extraction of unique information from program data without permanent modification of the program.

MATRIX$_x$ has greatly enhanced the potential for developing a powerful interactive model reduction program. The present program has demonstrated the feasibility of the approach, but is limited in the range of options available to the user. The present program modules are described, briefly, below. Complete source listings of all modules are contained in appendix B of the User's Guide.

1. MODRED

MODRED is the filename of the top-level command file which implements the model reduction process. It provides brief instruction to the user, explaining how to initiate program operation. It also controls the format of information displayed to the user during operation. MODRED transfers program control to the second-level command file MENU.

2. MENU

MENU is the filename of the second-level command file which displays the algorithm options available to the user and manages the flow of program execution to the desired options. At the completion of each option program control returns to MENU to provide the user an opportunity to execute a variation on the reduction just completed, to initiate an entirely different option, or to exit from the program.

3. LOAD

LOAD is the filename of the command file which loads a system model from a data file into the temporary data file SYSB.DAT for use in model reduction. The data file should contain an appropriate state variable model of a system in standard state-variable forms:

$$\dot{\underline{x}}(t) = [F]\underline{x}(t) + [G]\underline{u}(t) \ ;$$

and:

$$\underline{y}(t) = [H]\underline{x}(t) + [J]\underline{u}(t) \ ;$$

for continuous systems, or:

$$\underline{x}(k+1) = [F]\underline{x}(k) + [G]\underline{u}(k) \ ;$$

and:

$$\underline{y}(k) = [H]\underline{x}(k) + [J]\underline{u}(k) \ ;$$

for discrete-time systems. The models may represent either stable or unstable systems, but cannot contain poles at the origin (i.e. the system matrices must be invertable). A 59th order model has been run successfully (about fifty minutes of elapsed time), but a 75th order model caused an overflow in $MATRIX_x$. Discrete system models should include the sampling interval used in the model (for labeling the time axis in response plots). For SISO system models, the numerator and denominator coefficients may be included in the data file.

4. INDEX

INDEX provides a menu listing the demonstration system models and the various versions of each model normally available as part of the MODRED program modules. In addition, it describes the naming convention for data files and provides brief instructions for entering new models.

5. MOORE

MOORE is the filename of the command file which performs the SUBSYSTEM ELIMINATION algorithm (described in section D.4, above) for continuous systems. MOORE utilizes the command file BALANS described below to determine and perform the similarity transformations needed to "balance" the state-variable model. MOORE asks for user definition of a vector RCOL to indicate which subsystems are to be eliminated and which retained. It also asks for user definition of the variables NPTS and TS, the number of points and the time duration, respectively, of the step-response simulation.

MOORE displays a plot showing the step-response of the full system, the step-response of the reduced system, and the error function representing the difference between the two. If the system is a single-input single-output system, MOORE executes a command file (SING) which presents the user with the eigenvalues and the residues of the reduced system, and with the coefficients of the numerator and denominator polynomials of the reduced transfer function. If the system is a multivariable system, MOORE executes a command file (MULTI) which presents the user with the eigenvalues of the reduced system and the coefficients of the "observer canonical" state-variable model.

6. BALANS

BALANS is the filename of the command file which determines and performs the similarity transformations necessary to transform a continuous state-variable model into "balanced" form. If the system model is stable, the solution is based on the Grammian and is exact. If the system is unstable the solution is an approximation based on a short segment of the step-response. BALANS obtains the system model from a user-prepared data file SYSB. SYSB must contain the coefficients of the F, G, H, and J matrices in the usual state-variable model:

$$\underline{\dot{y}(t)} = \left[\ F\ \right]\underline{x(t)} + \left[\ G\ \right]\underline{u(t)}$$

$$\underline{y(t)} = \left[\ H\ \right]\underline{x(t)} + \left[\ J\ \right]\underline{u(t)}$$

SYSB is created in MATRIX$_x$ interactive mode. If the system

is unstable, BALANS will ask for user definition of the variables TI and NPTS, the time interval and number of points, respectively, to be used in the approximation.

BALANS displays the eigenvalues, the balanced model and the "Component Magnitudes" (second-order modes) of the system to assist the user in making the appropriate model reduction decisions. BALANS returns program control to the calling command file.

7. DISCRT

DISCRT is the filename of the command file which performs the SUBSYSTEM ELIMINATION algorithm (described in section D.4, above) for discrete-time systems. DISCRT utilizes the command file BALDISC described below to determine and perform the similarity transformations needed to "balance" the state-variable model. DISCRT asks for user definition of a vector RCOL to indicate which subsystems are to be eliminated and which retained. It also asks for user definition of the variables NPTS and TS, the number of points and the sampling interval, respectively, for the step-response simulation.

DISCRT displays a plot showing the step-response of the full system, the step-response of the reduced system, and the error function representing the difference between the two. If the system is a single-input single-output system, DISCRT executes a command file (SING) which presents the user with the eigenvalues and the residues of the reduced system, and with the coefficients of the numerator and denominator polynomials of the reduced transfer function. If the system is a multivariable system, DISCRT executes a command file (MULTI) which presents the user with the eigenvalues of the reduced system and the coefficients of the "observer canonical" state-variable model.

8. BALDISC

BALDISC is the filename of the command file which determines and performs the similarity transformations necessary to transform a discrete-time state-variable model into "balanced" form. The solution uses an approximation of the Grammian based on a short segment of the step-response. BALDISC obtains the system model from a user-prepared data file SYSB. SYSB must contain the coefficients of the F, G, H, and J matrices in the usual state-variable model:

$$\underline{y(k+1)} = \left[F \right] \underline{x(k)} + \left[G \right] \underline{u(k)}$$

$$\underline{y(k)} = \left[H \right] \underline{x(k)} + \left[J \right] \underline{u(k)}$$

SYSB can be created in MATRIX$_x$ interactive mode, but is usually loaded from a data file using the LOAD module. BALDISC will advise the user of the sampling interval, TI, specified in the model and ask for user definition of the variable NPTSG, the number of points to be used in the approximation of the Grammian. Guidance on the relationship between TI and NPTSG is provided in the User's Guide (pg 28).

BALDISC displays the eigenvalues, the balanced model and the "Component Magnitudes" (second-order modes) of the system to assist the user in making the appropriate model reduction decisions. BALDISC returns program control to the calling command file.

9. RTSEL

RTSEL is the filename of the command file which performs the ROOT SELECTION algorithm described above. RTSEL utilizes the command file SYSPRE described below to prepare an "observer canonical" state-variable model from a "transfer function" model. RTSEL asks for user definition of a vector RCOL to indicate which residues are to be retained and which deleted. It also asks for user definition of the variables NPTS and TS, the number of points and the time duration, respectively, of the step-response simulation.

RTSEL and SYSPRE display intermediate results needed for the user to make appropriate decisions in the reduction process. RTSEL displays a plot showing the step-response of the full system, the step-response of the reduced system, and the error function representing the difference between the two. RTSEL also presents the user with the coefficients of the reduced transfer function model and with the numerical value of the RMS error in the reduction. At the completion of the reduction, RTSEL returns program control to MENU.

10. RESREN

RESREN is the filename of the command file which performs the RESIDUE RETENTION algorithm described above. RESREN utilizes the command file SYSPRE described below to prepare an "observer canonical" state-variable model from a "transfer function" model. RESREN asks for user definition of a vector RCOL to indicate which residues are to be retained and which deleted. It also asks for user definition of the variables NPTS and TS, the number of points and the time duration, respectively, of the step-response simulation.

RESREN and SYSPRE display intermediate results needed for the user to make appropriate decisions in the reduction process. RESREN displays a plot showing the step-response of the full system, the step-response of the reduced system, and the error function representing the difference between the two.

RESREN also presents the user with the coefficients of the reduced transfer function model and with the numerical value of the RMS error in the reduction. At the completion of the reduction, RESREN returns program control to MENU.

11. SYSPRE

SYSPRE is the filename of the command file which prepares an "observer canonical" form state variable model from a "transfer function" model. SYSPRE asks the user for definition of two vectors AI and BI, the coefficients, respectively, of the denominator and numerator of the transfer function.

SYSPRE displays the eigenvalues and residues of the augmented system to assist the user in making appropriate model reduction decisions. SYSPRE returns program control to the calling command file.

12. DIFPN

DIFPN is the filename of the command file which performs the DIFFERENTIATION OF POLYNOMIALS algorithm described above. DIFPN asks the user to define the coefficients of the numerator and denominator coefficients of the transfer function model. It also asks for the order desired in the reduced model, and for definition of the variables NPTS and TS, the number of points and the time duration, respectively, of the step-response simulation.
DIFPN displays a plot showing the step-response of the full system, the step-response of the reduced system, and the error function representing the difference between the two. DIFPN also presents the user with the coefficients of the reduced transfer function model and with the numerical value of the RMS error in the reduction. At the completion of the reduction, DIFPN returns program control to MENU.

F. SUMMARY OF STUDY:

Indiana University - Purdue University at Indianapolis (the University) has completed research, during the period from December 1, 1983 through August 31, 1985, for the Model Reduction study described above. The effort consisted of 2.0 man-months during the period from May 15, 1984 through August 31, 1984 and 1.6 man months during the period from December 1, 1983 through May 14, 1984. The Principal Investigator for the study was Dr. Kenneth R. Dunipace.

The performance of the study required access to an interactive computer equipped with the MATRIX$_x$ programming language. The original plan for the University to provide access to a VAX 11/780 computer and to share in the cost of acquiring a copy of MATRIX$_x$

could not be achieved because of an unexpected change in licensing policy by the supplier. Instead, telephone/computer terminal access was provided to MATRIX$_x$ on the Ames VAX. Data communications over that distance were erratic, at best. Plans to use MATRIX$_x$ at West Lafayette were thwarted by installation delays until after the project had been terminated.

The research included literature review, algorithm and program development, and documentation. Each of these topic areas are discussed below. Within each area, topics are listed in order of priority. Progress in algorithm and program development was the principal activity. Effort in the other areas supported development progress.

1. LITERATURE REVIEW

References obtained during the summer of 1983 were reviewed in search of promising methods of model reduction. The emphasis was on methods which were directly applicable or could be readily adapted to multivariable systems. No alternative methods were found which appeared to be sufficiently promising to warrant inclusion within the current development.

2. ALGORITHM AND PROGRAM DEVELOPMENT

.....a) The algorithm for Balancing unstable systems was refined, verified, and evaluated. As a result, the Subsystem Elimination method was extended to include unstable continuous system models and both stable and unstable discrete-time models.

.....b) As described in section C above, additional Demonstration System Models were developed for use in evaluating model reduction methods. Experience has shown that some algorithms which work well with stable systems do not work well, if at all, with unstable systems. Thus, four unstable models were developed to verify and evaluate this aspect of algorithm performance.

Single-Input Single-Output models are important, in the development of algorithms, to give insight and experience in the behavior of the algorithm. Most applications, however, involve multivariable systems. Therefore, a set of six Multivariable Demonstration System Models were developed to supplement the single-input single-output models in verifying and evaluating algorithms applicable to multivariable systems.

.....c) The early coding of the program MODRED was performed while the principal investigator was learning MATRIX$_x$ language and, indeed, with an early version of MATRIX$_x$. In several areas the coding was inefficient, there were parts of

the coding which no longer perform an essential function, and there were inconsistencies in the form of documentation included. Much of the code was revised to improve efficiency and consistent documentation was incorporated in each module.

3. DOCUMENTATION

.....a) A User's Guide was prepared describing the use of the model reduction program MODRED and its subsidiary components. The User's Guide provides step-by-step instructions and examples for using the programs and limited background theory. The User's Guide was developed in parallel with the algorithm and program development of the pertinent program components and provides a level of background that is compatible with the present programs.
.....b) Tutorial material describing the theoretical background of model reduction in general and each of the methods incorporated in MODRED has been included in the User's Guide. The User's Guide also includes discussion of the Demonstration System Models and illustrates the typical strengths and weaknesses of the methods included in MODRED. These tutorial materials were prepared after a reasonably comprehensive version of MODRED and a functional draft of the User's Guide had been developed.

.....c) A condensed version of the User's Guide was incorporated into the program MODRED. These instructions are available as terminal displays accompanying the reduction by selecting the appropriate option from the program MENU.

.....d) A number of "error traps" and associated "help" instructions were incorporated in MODRED so that users have a reasonable opportunity to recover from mistakes.

G. RECOMMENDATIONS FOR FUTURE WORK

A number of the goals envisioned when this level-of-effort project began were not achieved. Some no longer seem relevant. Some have been modified and incorporate in the status section, F, above. The remainder are listed, as presented in the proposal, below:

.....a) Most of the theoretical work on model reduction appears to have used time-domain modeling methods. Since there is significant design and specification using frequency-domain modeling, special effort should be made to find at least one frequency-domain method of model reduction which can be applied to multivariable systems. Appropriate algorithms should be developed, verified, evaluated, and incorporated in MODRED.

.....b) The application of the program MODRED to design of a flight control system should be demonstrated. The intent was to use a linearized model of the X-29 for this demonstration.

.....c) In the literature, and throughout this study, Model Reduction is limited to operations on a mathematical model which produces a similar model of lower order. Identification provides an alternative method for determining low-order models for complicated systems, including the potential for obtaining a low-order linear model from a complicated nonlinear model. In the typical application, the input and output time-histories of a prototype system are recorded. The recordings are then processed mathematically to obtain a model with similar input-output characteristics. It is possible to produce comparable input-output records from a simulation of the complicated model of a system. Identification procedures could then be applied to the records to obtain a low-order model. This approach to model reduction is theoretically possible. It should be investigated for practical feasibility.

H. BIBLIOGRAPHY

1. S. C. Shah, R. A. Walker, and C. Z. Gregory jr.; "MATRIX$_x$ User's Guide", Integrated Systems, Inc., Palo Alto, Ca., 1982

2. R. A. Walker and N. Gupta; "Flight Test Trajectory Control Analysis", Integrated Systems, Inc., Palo Alto, Ca., 1982

3. B. C. Moore; "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction.", IEEE Transactions on Automatic Control, Vol. AC-26, No. 1 Feb. '81, pp. 382-387.

4. R. C. Dorf; "Modern Control Systems", third edition, Addison-Wesley, 1980.

5. Per-Olof Gutman, Carl Frederick Mannerfelt, and Per Molander; "Contributions to the Model Reduction Problem", IEEE Transactions on Automatic Control, Vol. AC-27, No. 2, Apr. '82, pp. 454-455.

MODEL REDUCTION METHODS FOR CONTROL DESIGN

Appendix A

Figures

Comparative Performance of Model Reduction Algorihms

(RMS Error x 104)

NASA
DFRF83-1114

Reduction method	System model		
	#1	#2	#3
Root Selection	9.6	3029	669
Residue retention	9.7	391	669
Different of polynomials	1000	780	980
Subsystem elimination	1.3	2.9	2.3

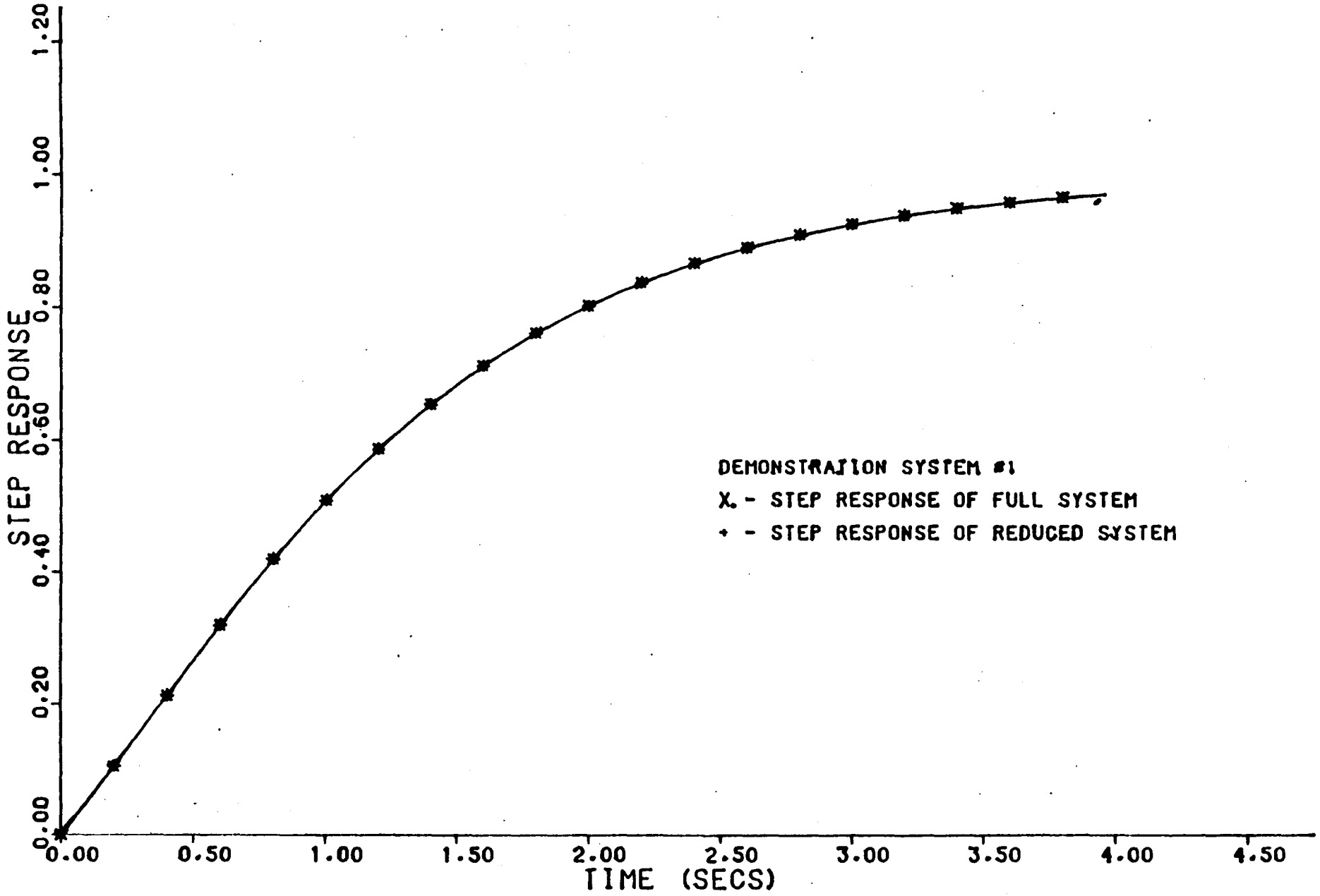


FIGURE 2. MODEL REDUCTION BY ROOT SELECTION

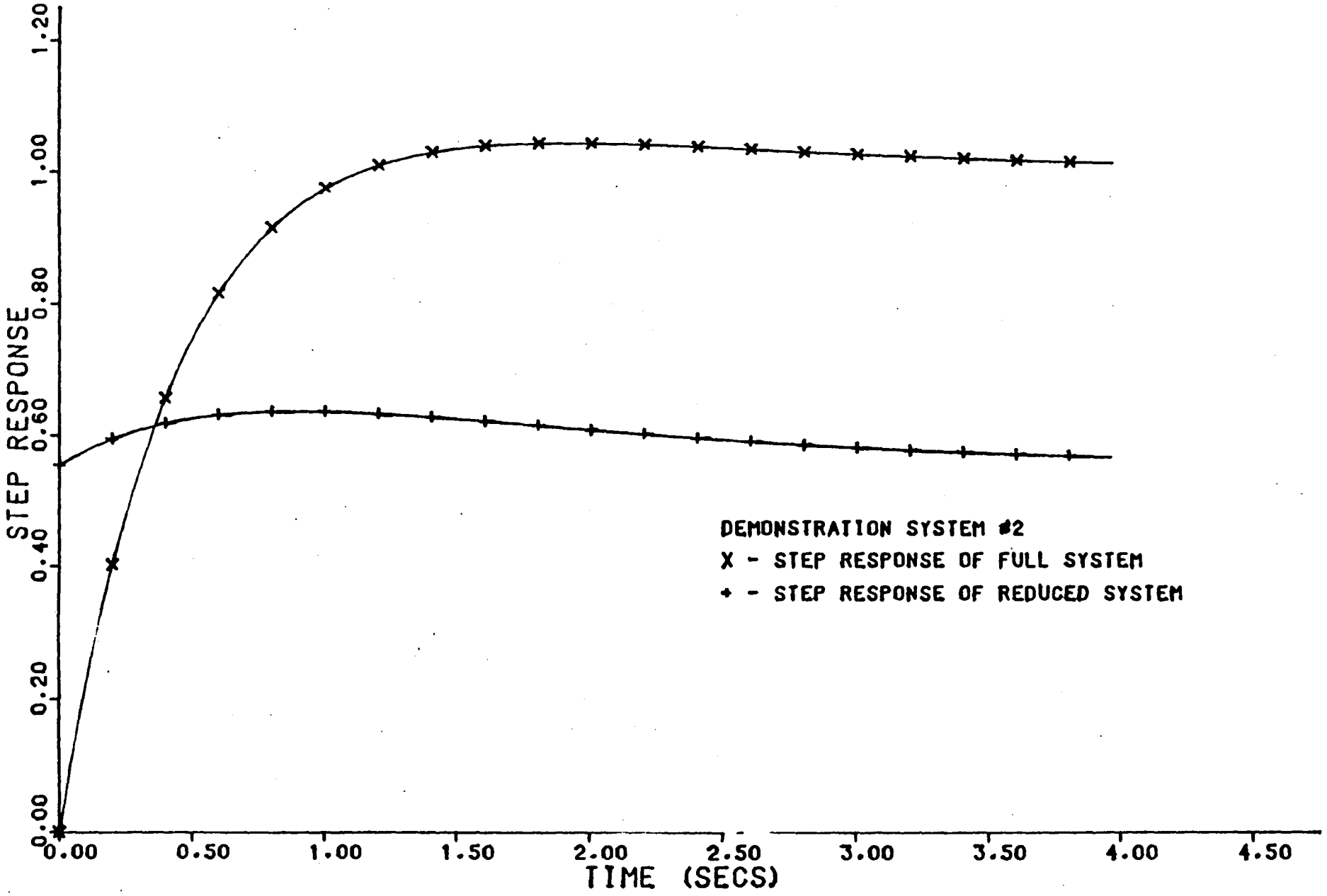


FIGURE 3. MODEL REDUCTION BY RESIDUE RETENTION

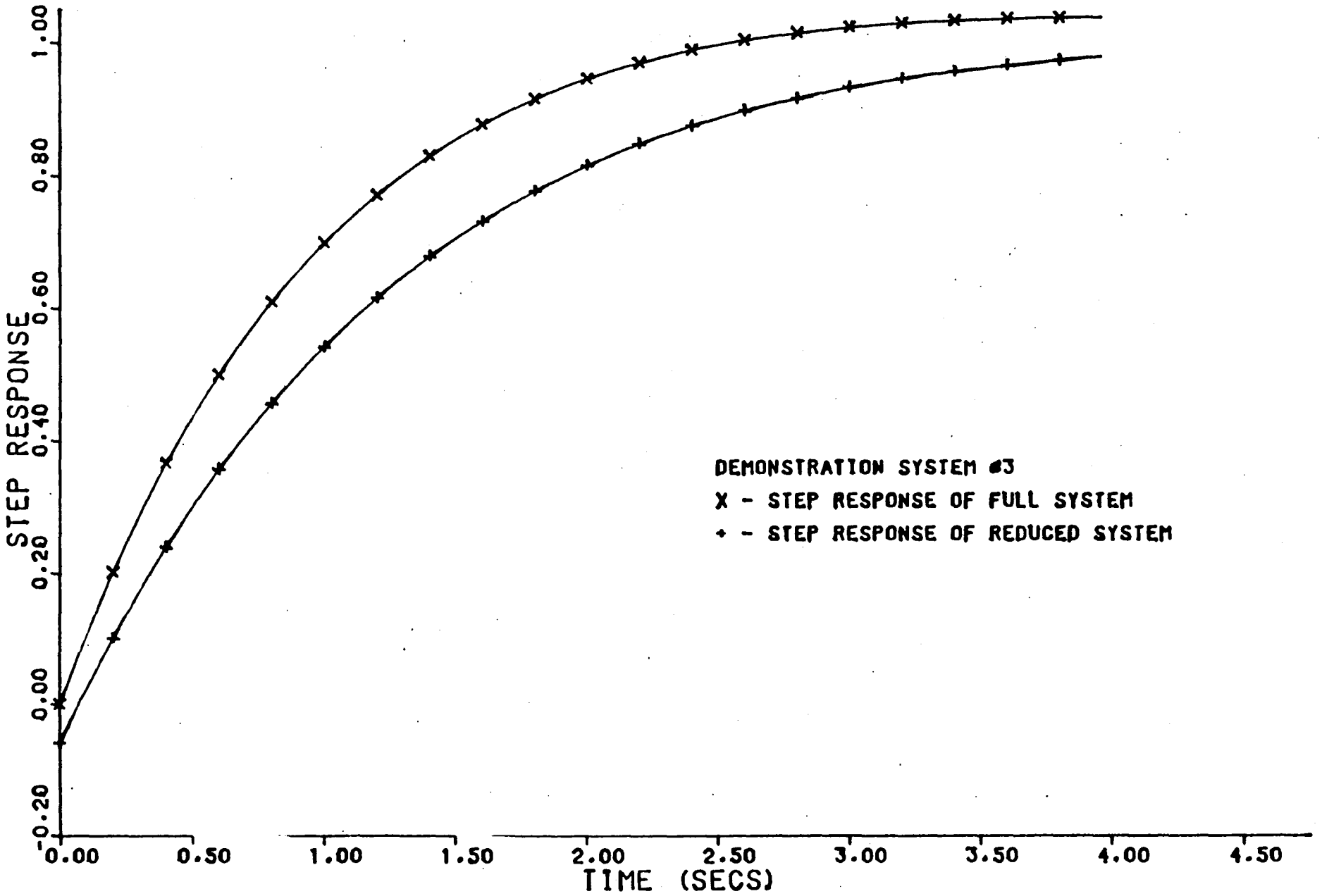


FIGURE 4. MODEL REDUCTION BY DIFFERENTIATION OF POLYNOMIALS

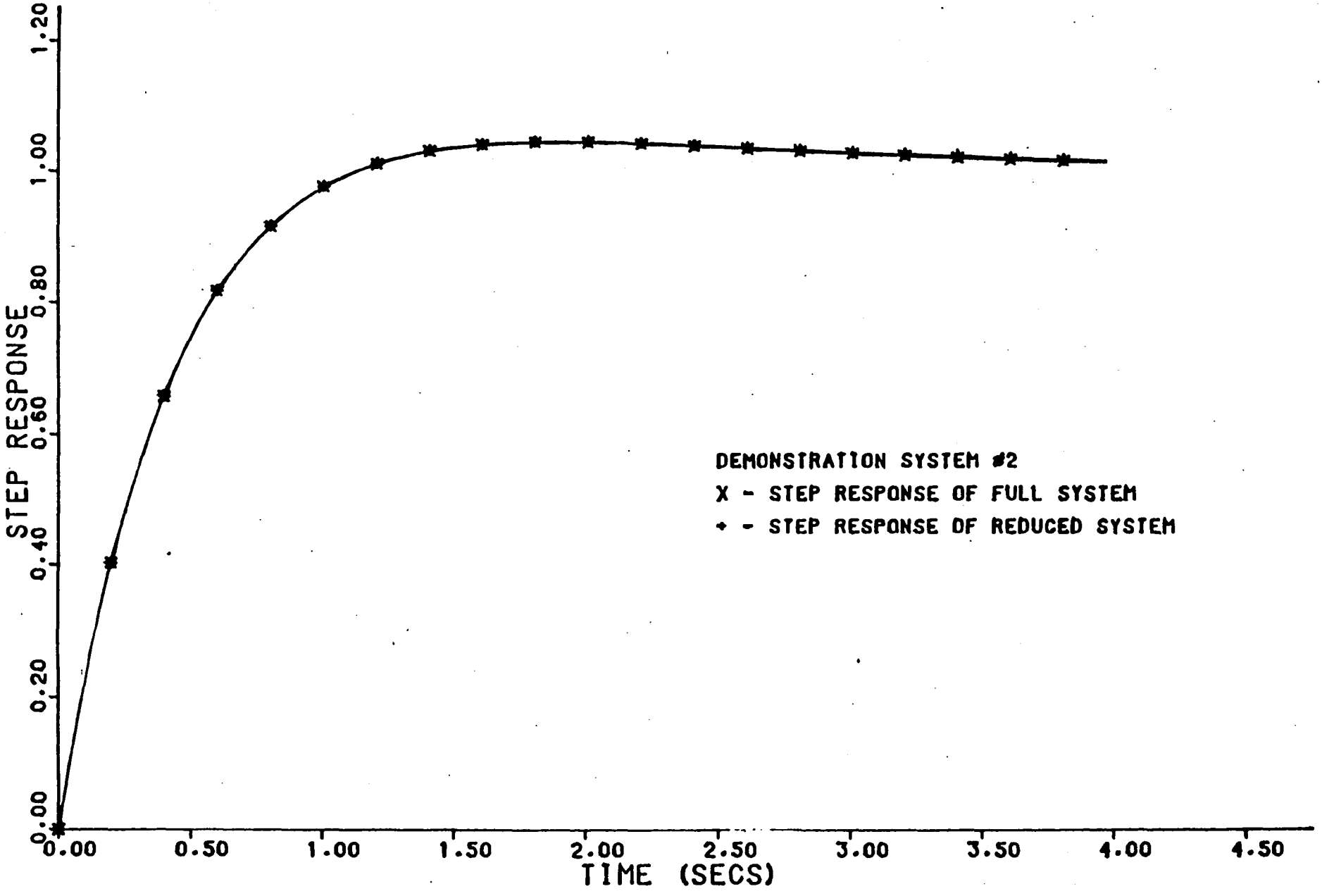


FIGURE 5. MODEL REDUCTION BY SUBSYSTEM ELIMINATION

MODEL REDUCTION METHODS FOR CONTROL DESIGN

Attachment

MODRED USER'S GUIDE

Model Reduction Methods for Control Design

Prepared for:

National Aeronautics and Space Administration

Dryden Flight Research Facility

Edwards, California

by

Dr. K. R. Dunipace

Engineering Division

Indiana University-Purdue Univeristy at Indianapolis

1985

MODRED USER'S GUIDE
TABLE OF CONTENTS

MODRED USER'S GUIDE

5/25/85

1. Introduction:

"MODRED" is an interactive program written in the MATRIX$_x$ programming language. It offers five different algorithms for model reduction of linear dynamic systems. Three of the algorithms apply to continuous-time single-input single-output systems represented by transfer functions. One of the algorithms applies to continuous-time systems represented by a state-variable model. The final algorithm applies to discrete-time systems represented by a state-variable model. Systems represented by state-variable models may be multi-input multi-output.

In addition to the algorithms for model reduction, the program contains brief instructions for its operation, a built-in demonstration, a set of system models for displaying and studying the behavior of the different algorithms, an index of the system models, and a program module to load state-variable models.

Model reduction is viewed as an early step in the control design process. At this stage, the designer is gaining understanding of and familiarity with the system. Thus, the program is very interactive, providing information about the significant characteristics of the system, waiting for operator evaluation and decisions, and finally describing the characteristics of the reduced model.

The experienced user is offered an option of omitting most of the prompting messages.

2. Organization of User's Guide:

The first three chapters of this User's Guide provide general background on the program MODRED and the User's Guide itself. Chapters 4 through 9 contain instructions and examples for operating the program and performing the several tasks necessary to obtain results. Chapters 10 through 13 provide brief theoretical explanation for each of the model reduction techniques available through the MODRED program. The appendices contain the figures, program listings, models of systems used in the examples,

and the bibliography.

System models provided with MODRED all have filenames of the form SYS*.DAT. The asterisk is replaced by system identification. Systems which were created as part of program development and testing are numbered (presently) from one to nine. The basic identifier (e.g. SYS1) indicates a transfer function model stored for reference (in version 4.0 of MODRED transfer function models cannot be loaded directly into reduction algorithms). A suffix S is added for state-variable models (e.g. SYS1S), and a suffix D is added for discrete-time models (e.g. SYS1SD). There are no discrete-time transfer function models. Other system models which have been used for demonstration purposes identify the modeled system (i.e. F4, F15, X29).

Throughout this User's Guide text will be typed in this type style ("Roman"), displays presented by the program will be typed in **this type style ("Bold")**, while user entries to the program will be typed in *this type style ("Italic")*. All command lines end with a "carriage return". They are not shown in the examples because they will not appear on the terminal display.

3. Requirements:

Access to MATRIX$_X$ software produced by Integrated Systems Inc. (version 4.0, or later) is required to use the program MODRED. Instructions in this manual assume that MATRIX$_X$ Version 4.0 is installed in the computer being used. Examples were developed using the Research Engineering Division VAX/VMS at NASA/Dryden Flight Research Center. Operating system commands may be different on other machines. MATRIX$_X$ should be independent of the machine. In addition, the user should have a basic background in continuous and digital control theory, as presented, for example, by Dorf in "Modern Control Systems" (ref. 1) and by Franklin and Powell in "Digital Control" (ref. 2).

4. Program Initiation:

Obtain the operating system prompt (a dollar sign $ in the VAX/VMS operating system and throughout the examples in this manual) following normal login procedures.

Initiate MATRIX$_X$:

```
$ mat40
```

MATRIX$_X$ will respond with a message similar to:.

Enter terminal type: 1 = TTY
2 = Lineprinter
3 = VT125
4 = Tektronix 4014

The terminals actually listed will vary with the installation. Select the most appropriate option. If all else fails try TTY. (A Tektronix 4014 was used for this demonstration.)

4

```
**************************************
*                                    *
*              (TM)                  *
*      MATRIX        V4.0            *
*            X                       *
*                                    *
* (C) Copyright 1982,1983,1984       *
*   Integrated Systems Inc.          *
**************************************
```

HELP is available

<>

The <> symbol is the $MATRIX_X$ prompt (also used as the logical expression "not equal to").

Now initiate MODRED:

<> *exec('modred')*

MODRED will respond with an initial logo:

```
1234567890123456789012345678901234567890123456789012345678901234567890123456
2                                                                           7
3                                                                           8
4                                                                           9
5                                                                           0
6     ************************************************************        1
7     *                                                          *        2
8     *                        MODRED V4.0                       *        3
9     *                           BY                             *        4
0     *                      K. R. DUNIPACE                      *        5
1     *   INDIANA UNIVERSITY-PURDUE UNIVERSITY-INDIANAPOLIS      *        6
2     *                         8/7/85                           *        7
3     *                                                          *        8
4     ************************************************************        9
5                                                                           0
6                                                                           1
7                        DEVELOPMENT OF                                     2
8          THIS PUBLIC DOMAIN PROGRAM WAS SPONSORED BY                      3
9             NASA - DRYDEN FLIGHT RESEARCH FACILITY                        4
0                                                                           5
1234567890123456789012345678901234567890123456789012345678901234567890123456
```

After a few moments, MODRED will present the first menu and options:

MODRED IS AN INTERACTIVE MODEL REDUCTION PROGRAM.

THIS VERSION WORKS WITH MATRIXX VERSION 4.0.

ENTER THE NUMBER OF THE DESIRED OPTION.

1) INSTRUCTIONS
2) NOVICE
3) EXPERT

MODRED will automatically assign a NOVICE classification after the instructions have been read, when the INSTRUCTIONS option is selected. Most users will skip the instructions and select the NOVICE option. That case will be presented here.

A fold-out flow chart of MODRED, included as figure 1, is inserted at the back of the User's Guide so that the user can follow program flow while working through the following examples.

ENTER DESIRED OPTION *2*

MODRED will next display a menu, largely of available model reduction algorithms, and prompt for a choice:

ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).

1 DOMINANT ROOT SELECTION

2 RESIDUE RETENTION

3 DIFFERENTIATION OF POLYNOMIALS

4 "MOORE" REDUCTION - CONTINUOUS

5 "MOORE" REDUCTION - DISCRETE

6 SYSTEM MODELS INDEX

7 LOAD A SYSTEM MODEL

ENTER SELECTED OPTION NUMBER

**** WARNING: Do not respond to this prompt at this time! ****

This completes Program Initiation.

Note: If the user makes an error (such as typing a letter instead of a number as an option) MODRED will prompt:

SELECTED OPTION NOT AVAILABLE. TRY AGAIN.

5. DEMONSTRATION REDUCTION (MANUAL ENTRY):

This section contains a simple exercise showing how to use the first three algorithms in the menu above. Each of these operate on Transfer Function Models of a system (after obtaining model definition interactively from the terminal). Method #1 "Dominant Root Selection" is used for this exercise. The other two methods follow almost identical procedures and contain a built-in demonstration (as do all the algorithms) to help the user through a sample reduction. In the following exercise the user will perform the built-in demonstration contained in algorithm #1 "Dominant Root Selection".

In this demonstration, the system model is:

$$T(s) = \frac{0.48(S+5)^2}{(S+1)(S+2)(S+6)} = \frac{0.48S^2+4.8S+12}{S^3+9S^2+20S+12}$$

Now, respond to the prompt left "open" at the end of Program Initiation:

ENTER SELECTED OPTION NUMBER *1*

This selection turns control of the reduction over to program

module RTSEL, which in turn initiates program module SYSPRE. SYSPRE asks for the coefficients of the transfer function:

This process sometimes takes several seconds (seems longer), so MENU assures the user that the computer is, in fact, progressing:

*** * * * * COMPUTING. PLEASE WAIT. * * * * ***

This note will appear from time-to-time at points in the various programs where experience has shown that computation is sometimes slow. The note will be repeated in the demonstrations in this User's Guide, but without further explanation. This note is suppressed in expert mode.

Similarly, there are various points in the program where data is displayed for user evaluation. At these points the program is halted and the following note is displayed to remind the user that the computer is not progressing. Enter a "carriage return" to resume computation.

*** * * * * WAITING. CR TO CONTINUE. * * * * ***

PAUSE>

This note will be repeated in the demonstrations without further comment. This note is suppressed in expert mode.

ENTER COEFFICIENTS OF NUMERATOR POLYNOMIAL.

E.G. <0.48,4.8,12>

ENTER BI *<0.48,4.8,12>*

ENTER COEFFICIENTS OF DENOMINATOR POLYNOMIAL.

E.G. <9,20,12>

N.B. COEFF. OF "SN" (NOT ENTERED) MUST BE 1.**

ENTER AI *<9,20,12>*

SYSPRE will now do the calculations involved in determining the eigenvalues of the system and the residues of the system step response. On the basis of these the user will decide which roots are non-dominant and may be discarded without drastically altering the system performance. The step-response is represented by an eigenvalue at the origin. It must be retained for proper program performance. Unstable eigenvalues should be retained. Similarly, stable eigenvalues with large residues contribute heavily to system performance. SYSPRE displays the eigenvalues and residues for evaluation and decision making, then returns control to RTSEL

for the actual reduction.

```
THE EIGENVALUES OF THE AUGMENTED SYSTEM ARE:

 IGN          =

          -6.0000  + 0.0000i
          -2.0000  + 0.0000i
          -1.0000  + 0.0000i

  * * * * * WAITING. CR TO CONTINUE.  * * * * *

  PAUSE>
```

The system is augmented (as in section 4.3.1 of MATRIX$_x$ User's Guide) (ref. 3) to provide a step response instead of an impulse response. Record the location of all (if any) unstable eigenvalues in the list. They should be selected (retained) regardless of the size of their residues.

SYSPRE now calculates and displays the residues which are to be used in the selection process.

```
THE RESIDUES OF THE AUGMENTED SYSTEM ARE:
RES          =
          -0.0040  + 0.0000i
           0.5400  + 0.0000i
          -1.5360  + 0.0000i
           1.0000  + 0.0000i

  * * * * * WAITING. CR TO CONTINUE.  * * * * *

  PAUSE>
```

Program control is now returned to RTSEL for the actual model reduction process. RTSEL will ask you to indicate the eigenvalues (roots) to be selected for the reduced model.

```
SELECT RESIDUES TO BE RETAINED.

ENTER, IN ORDER, 0 OR 1 FOR EACH RESIDUE.

"1" RETAINS RESIDUE; MUST RETAIN LAST RESIDUE.

 E.G.:   <0,1,1,1>

ENTER RCOL <0,1,1,1>

  * * * * * COMPUTING. PLEASE WAIT.  * * * * *
```

RTSEL will now calculate and display the state-variable model of the reduced/augmented system:

```
THE REDUCED/AUGMENTED MODEL IS:

FR          =
        -2.0000      0.0000      0.0000
         0.0000     -1.0000      0.0000
         0.0000      0.0000      0.0000

GR          =
        1.
        1.
        1.

HR          =
0.5400     -1.5360      1.0000

* * * * * WAITING. CR TO CONTINUE.  * * * * *

 PAUSE>
```

RTSEL will then ask for the time duration and number of points to be used in simulating the step responses. They are plotted for comparing the performance of the full and reduced system. In selecting these parameters, the user should observe the usual restrictions imposed in sampling continuous functions. In model reduction, a time duration (T) of four times the longest time constant (i.e. four divided by the smallest real part of a non-zero stable eigenvalue) is usually adequate to display the slow characteristics of the systems. Similarly, the number of points should be sufficient to adequately display the fast characteristics of the system. This can usually be achieved with a number of points equal to the ratio of the time duration to the shortest time constant (i.e. the time duration times the largest real part of a non-zero stable eigenvalue). In this sample case the smallest real part of a non-zero eigenvalue is 1.0 while the largest is 6.0. Hence, the time duration should be 4.0/1.0 = 4.0 seconds (the example actually uses only 3.5 seconds, and the resulting plots do not quite reach steady-state). Similarly, the number of points should be 4.0*6.0 = 24 (again, the example doesn't quite meet this criteria).

```
ENTER TIME DURATION FOR SIMULATION.  E.G. 3.5

ENTER T 3.5

ENTER NUMBER OF POINTS FOR SIMULATION.  E.G. 20

ENTER NPTS 20
```

*** * * * * COMPUTING. PLEASE WAIT. * * * * ***

RTSEL prepares the data representing the step responses of both the full system and the reduced system and the error function representing the difference between the full system response and the reduced system response. The data is then plotted at the terminal. A carriage return will abort the plot, so RTSEL warns the user to wait for the plot (it is quite slow).

— WHEN FINISHED WITH THE PLOT, NOT NOW, CR TO CONT. —

In this example the reduction is so good that the plot (figure 2) barely shows the difference (in fact, it will show no difference at all on a non-graphics terminal). System model #2 cannot be reduced effectively by this method, hence the plot produced in that case is much more instructive. Try it, later, as an independent model reduction exercise.

RTSEL will now calculate and display the RMS error caused by the reduction process.

THE RMS ERROR FOR THIS REDUCTION IS:

ERR =

9.5480D-04

RTSEL will now display the numerator and denominator coefficients for the reduced transfer function (the step-augmentation is removed before these coefficients are determined). The coefficients are presented from highest order to zero-order for both numerator and denominator.

IN THE REDUCED TRANSFER FUNCTION,

THE COEFFICIENTS OF THE NUMERATOR ARE:

NUM =
0.0040 - 0.0000i
0.4680 - 0.0000i
2.0000 - 0.0000i

THE COEFFICIENTS OF THE DENOMINATOR ARE:

DEN =
1.0000 + 0.0000i
3.0000 - 0.0000i
2.0000 - 0.0000i

*** * * * * WAITING. CR TO CONTINUE. * * * * ***

PAUSE>

This corresponds to the transfer function:

$$T(s) = \frac{0.004S^2+0.468S+2}{S^2+3S+2}$$

This is the end of this reduction process. RTSEL now returns control to MENU for the same options as were presented in section 4. Program Initiation, above, i.e.:

ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).

1 DOMINANT ROOT SELECTION

2 RESIDUE RETENTION

3 DIFFERENTIATION OF POLYNOMIALS

4 "MOORE" REDUCTION - CONTINUOUS

5 "MOORE" REDUCTION - DISCRETE

6 SYSTEM MODELS INDEX

7 LOAD A SYSTEM MODEL

ENTER SELECTED OPTION NUMBER

6. DEMONSTRATION REDUCTION (DATA FILE ENTRY):

This section contains a simple exercise showing how to use the two algorithms (4, and 5 in the menu above) which operate on State Variable Models of a system (and obtain model definition from a data file "SYSB"). Method #5 "Moore" Reduction - Discrete is used for this exercise. The other method (#4 "Moore" Reduction - Continuous) uses almost identical procedures and contains a built-in demonstration (as do all the algorithms) to help the user through a sample reduction using that algorithm. In this exercise the user will perform the demonstration contained in algorithm #5 "Moore" Reduction - Discrete. A system model is loaded into data file SYSB.DAT before beginning the reduction procedure.

Complete the instructions contained in Section 4. "Program Initiation", above.

Now, respond to the prompt left "open" at the end of Program Initiation:

ENTER SELECTED OPTION NUMBER *7*

*** * * * * COMPUTING. PLEASE WAIT. * * * * ***

This selection turns control over to program module LOAD to load a system model into the file SYSB and onto the stack. The system model is loaded from a storage file designated by the user (*filename* in Fig. 1). The file for this example already exists (as SYS1D.DAT). A later section (sec. 8) of this User's Guide explains the creation of data files for user system models. The data stored in SYSB is used by MODRED in the reduction process. The data loaded onto the stack is a preliminary step in loading SYSB. LOAD initially clears the stack, so the contents of the stack at the end of the loading process will be the contents of SYSB only. Similarly, in the process of SAVEing SYSB, LOAD will clear any data which may have been previously stored in SYSB.

LOAD provides instructions:

> **THIS BRANCH OF "MODRED" LOADS A SYSTEM MODEL. THE MODEL IS LOADED ONTO THE STACK AND INTO A FILE NAMED 'SYSB'. 'SYSB' IS USED BY METHODS USING STATE-VARIABLE MODELS. THE STATE-VARIABLE MOBEL SHOULD DEFINE F,G,H,J, AND TS IN THE USUAL STATE-VARIABLE MODEL (TS IS THE SAMPLING INTERVAL, WHERE APPLICABLE).**
>
> *** * * * * WAITING. CR TO CONTINUE. * * * * ***
>
> **PAUSE>**
>
> **THE STACK IS USED BY SOME SISO (TRANSFER FUNCTION) METHODS. IF SUCH APPLICATIONS ARE PLANNED, THE DATA FILE SHOULD ALSO CONTAIN A VECTOR 'BI' CONTAINING THE COEFFICIENTS OF THE NUMERATOR POLYNOMIAL, AND A VECTOR 'A1' CONTAINING THE COEFFICIENTS OF THE DENOMINATOR POLYNOMIAL. THE DENOMINATOR POLYNOMIAL SHOULD BE NORMALIZED WITH A LEADING COEFFICIENT (HIGHEST POWER OF S) OF ONE. THE ONE IS NOT INCLUDED IN 'AI'. ENTER THE NAME OF THE FILE CONTAINING THE SYSTEM MODEL. THE FILENAME MUST BE ENCLOSED IN APOSTROPHES. E.G. 'FILENAME'**
>
> **ENTER FILENAME** *'SYS1D'*

Note: 'SYS1D' is a discrete-time state-variable version of the transfer function model used in the exercise in Section #5 Demonstration Reduction (Manual Entry) above.

LOAD will tell the user that the loading is completed and then return control to MENU.

-- LOAD COMPLETED --

MENU will again display the available choices. This time select reduction method #5 "Moore Reduction" - Discrete.

ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).

1 DOMINANT ROOT SELECTION

2 RESIDUE RETENTION

3 DIFFERENTIATION OF POLYNOMIALS

4 "MOORE" REDUCTION - CONTINUOUS

5 "MOORE" REDUCTION - DISCRETE

6 SYSTEM MODELS INDEX

7 LOAD A SYSTEM MODEL

ENTER SELECTED OPTION NUMBER *5*

* * * * * COMPUTING. PLEASE WAIT. * * * * *

This selection turns control of the reduction over to program module DISCRT, which in turn initiates program module BALDISC. BALDISC loads SYSB, determines and displays the eigenvalues of the system. The system model is:

F =

0.1149	0.0786	0.0086
-1.6748	0.8224	0.1556
-0.9433	-0.1027	0.9935

G =

0.0853
0.8225
1.9163

H =

1.0	0.0	0.0

J =0.0

TS =0.1667

```
THE EIGENVALUES OF THE SYSTEM ARE:

 IGN        =
          0.3679  + 0.0000i
          0.7165  - 0.0000i
          0.8465  + 0.0000i

  * * * * * WAITING. CR TO CONTINUE.  * * * * *

  PAUSE>
```

BALDISC will now remind the user that the system model is being processed as a discrete-time model, and ask for user selection of the number of points to be used in approximating the Grammian. In selecting this parameter, the user should observe the usual restrictions imposed in discrete-time representation of continuous functions. If the real parts of any complex eigenvalues are negative or if any real eigenvalues are smaller than +0.4, the sampling rate is probably too slow for the system. If the imaginary parts of all the complex eigenvalues are less than 0.5 or the real parts of all the eigenvalues are greater than +0.8 then the sampling rate is probably too fast for the system. In model reduction, a time duration of two to three times the longest time constant is usually adequate to measure the slow characteristics of the systems. This can be achieved by selecting an integer approximately equal to 2.0/(-ln(r)), where r is the largest radial distance to a stable eigenvalue (for this application integrators at +1.0 should not be considered stable). Similarly, model reduction must measure the fast characteristics of the system. This has to be achieved through selection of sampling interval. In this simple example, there are no complex eigenvalues, while the real eigenvalues range from +0.55 to +0.9. Thus, the sampling rate is probably satisfactory. The largest radial distance is 0.9, so the number of points should be a little larger than 2.0/(-ln(0.9) = 18.98. Hence, the number of points should be about 19 (the example uses 20).

```
THIS SYSTEM IS DISCRETE, THUS THE GRAMMIAN IS ONLY A
FINITE-TIME APPROXIMATION.  DEFINE SAMPLING POINTS FOR
APPROXIMATING GRAMMIAN.  E. G. 20
THE SAMPLING INTERVAL IS:
TS = 0.1

ENTER NUMBER OF POINTS NPTSG 20

  * * * * * COMPUTING. PLEASE WAIT.  * * * * *
```

BALDISC will now perform a number of calculations required in the reduction process, including a discrete-time approximation of the "balancing" algorithm presented by B. C. Moore in "Principal Component Analysis in Linear Systems", IEEE AC-26, 1-81, pp21ff (ref. 4). This algorithm is the principle element in both "Moore" reduction methods. BALDISC displays the resulting model. In a single-input single-output system the [F] matrix should be magnitude-symmetric and the G and H vectors should be magnitude equivalent. Multi-variable systems should approximate these characteristics. Observation of the balanced model provides the user a qualitative indication of how well the balancing algorithm succeeded.

THE BALANCED MODEL IS:

```
FB        =
        0.9015    -0.0995     0.0059
        0.0995     0.6711     0.0569
        0.0059    -0.0569     0.3583

GB        =
     0.3259
    -0.1451
    -0.0102

HB        =
     0.3259      0.1451    -0.0102
```

BALDISC will now display the component magnitudes. The component magnitudes serve a function similar to the residues in the Root Selection method above. That is, they show the importance of various input/state-variable/output combinations in producing the various impulse responses of the system. The user should evaluate the component magnitudes and decide which seem insignificant relative to the others. These should be discarded in the next step.

THE COMPONENT MAGNITUDES ARE:

```
LAM       =
        3.2322D+01
        1.0875D+00
        1.4219D-03
```

BALDISC now returns program control to DISCRT for the actual model reduction process. DISCRT will ask you to indicate the components representing the subsystems to be selected for the reduced model.

SELECT COMPONENTS TO BE RETAINED.
ENTER, IN ORDER, "0" OR "1" FOR EACH COMPONENT.
"1" RETAINS COMPONENT; "0" DELETES.
E. G.: <1,1,0>

ENTER RCOL *<1,1,0>*

DISCRT will now ask for the number of points to be used in simulating the impulse responses of both the full and reduced systems. This need not be the same as used for approximating the Grammian though the same considerations apply.

ENTER NO. OF POINTS FOR SIMULATION. E.G.: 20

ENTER NPTS *20*

* * * * * COMPUTING. PLEASE WAIT. * * * * *

DISCRT will now reduce the system and prepare the step response data for display. The step response data includes the step responses of both the full system and the reduced system and the error functions representing the difference between the full system response and the reduced system response. For multi-variable system the responses include all the separate combinations of input variable to output variable. The data is plotted at the terminal. A carriage return will abort a plot, so DISCRT warns the user to wait for the plot (it is quite slow, and for a multi-variable system there may be many).

— WHEN FINISHED WITH PLOT, NOT NOW, CR TO CONT. —

In this example, the reduction is so good that the plot (figure 3) shows no difference between the full and reduced system. It is, nevertheless, significantly better than the reduction achieved above by Root Selection. The improvement may be measured by comparing the RMS error of the two procedures. Unlike Root Selection, this algorithm is quite effective in reducing system with equal residues, such as system model #2. Try it later and notice how much better it works than Root Selection.

DISCRT will now calculate and display the RMS error caused by the reduction process (in a multi-variable case, it is the average of all the responses).

THE RMS ERROR FOR THIS REDUCTION IS:

ERR =

3.7311D-04

DISCRT will now display the reduced system model. For a multi-variable system the model will, of course, be a state-variable model. For this example system, and other single-input single-output systems the model is displayed as the coefficients of the transfer function. The program module SING provides this display, while the program module MULT provides the multi-variable display.

SING first displays the eigenvalues (they will virtually never be any of those from the full system) and the residues of the impulse response for the reduced system:

```
     THE EIGENVALUES OF THE REDUCED SYSTEM ARE:

      IGNOI        =
             0.7282
             0.8445

     THE RESIDUES OF THE REDUCED SYSTEM ARE:

      RESI         =
            -0.1644
             0.2495

       * * * * * WAITING. CR TO CONTINUE.   * * * * *

       PAUSE>
```

SING will now display the coefficients of the transfer function.

```
          IN THE REDUCED TRANSFER FUNCTION, THE COEFFICIENTS OF
          THE NUMERATOR POLYNOMIAL ARE (NUM. PN. MAY NOT BE
          CORRECT IF RESIDUES ARE COMPLEX.)

               GO           =
                      0.0852
                     -0.0429

          THE COEFFICIENTS OF THE DENOMINATOR POLYNOMIAL ARE:

               PDEN         =
                      1.0000
                     -1.5726
                      0.6149

        * * * * * WAITING. CR TO CONTINUE.   * * * * *
```

PAUSE>

This is the end of the reduction process. SING will now return control to DISCRT. DISCRT will return control to MENU. MENU will offer the same options as were presented in section 4. Program Initiation, above, i.e.:

ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).

1 DOMINANT ROOT SELECTION

2 RESIDUE RETENTION

3 DIFFERENTIATION OF POLYNOMIALS

4 "MOORE" REDUCTION - CONTINUOUS

5 "MOORE" REDUCTION - DISCRETE

6 SYSTEM MODELS INDEX

7 LOAD A SYSTEM MODEL

ENTER SELECTED OPTION NUMBER

7. CREATING A TRANSFER FUNCTION MODEL:

Transfer function models are entered into MODRED interactively in response to prompts from the program. Thus, 'creation' of a transfer function model is, essentially, the paper-and-pencil process of writing down the coefficients of the transfer function so they may be entered in response to prompts. For example, the transfer function (SYS1):

$$G(s) = \frac{0.48S^2+4.8S+12.0}{S^3+9.0S^2+20.0S+12.0}$$

Requires two coefficient vectors for reduction:

BI = [0.48,4.8,12.0], and

AI = [9.0,20.0,12.0].

Note that the transfer function must be normalized so that the coefficient of S^n is 1.0. Note also, that the normalized coefficient is not entered in the vector. MATRIX$_x$ will accept integer representation of real numbers (see demonstration in section 4, above).

A transfer function model may be saved for record and reference. Obtain the MATRIXX$_x$ prompt <> by following the instructions in

Section 4, Program Initiation (from the system operating system) or by selecting option 0 from the main option list in MENU (from MODRED).

Now enter the model onto the stack (MATRIX$_x$ will repeat back each entry):

```
<>BI = [0.48,4.8,12.0]

BI  =
   0.48  4.8  12.0

<>AI = [9.0,20.0,12.0]

AI  =
   9.0  20.0  12.0
```

Save the model to a data file (in this example SYSX even though the model is really SYS1):

```
<>SAVE('SYSX')
```

The model may be recovered by the command:

```
<>LOAD('SYSX')
```

8. CREATING A STATE-VARIABLE MODEL:

State-variable models are entered automatically from the file SYSB.DAT into MODRED as a program function. They must be created in advance of the reduction process and loaded into SYSB.DAT using the program module LOAD as described in Section 6, Demonstration Reduction (Data File Entry) above. For example, the discrete-time state-variable model (SYS1D):

$$\underline{x(k+1)} = \begin{bmatrix} 0.2249 & 0.0786 & 0.0086 \\ -1.6748 & 0.8224 & 0.1556 \\ -0.9433 & -0.1027 & 0.9935 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.0853 \\ 0.8225 \\ 1.9163 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = [1.0\ 0.0\ 0.0]\underline{x(k)} + [0.0]\underline{u(k)}$$

Requires a data file with four matrices ([F], [G], [H], and [J])

and a sampling interval (TS) for reduction.

The suggested procedure for creating such a file is demonstrated below.

Obtain the MATRIXX$_x$ prompt <> by following the instructions in Section 4, Program Initiation (from the system operating system) or by selecting option 0 from the main option list in MENU (from MODRED).

Now enter the model onto the stack (MATRIX$_x$ will repeat back each entry):

```
<>F=[0.2249 0.0786 0.0086
-1.6748 0.8224 0.1556
-0.9433 -0.1027 0.9935]

 F          =
        0.1149     0.0786     0.0086
       -1.6748     0.8224     0.1556
       -0.9433    -0.1027     0.9935

<>G=[0.0853
0.8225
1.9163]

 G          =
        0.0853
        0.8225
        1.9163

<>H=[1 0 0]

  H    =

     1.      0.      0.

<>J=0

  J    =

     0.

<>TS=0.1667

  TS   =

     0.1667
```

Save the model to a data file (in this example SYSX even though the model is really SYSID):

<>SAVE('SYSX')

The model is now created and stored. It may be loaded for system reduction following the procedures described in Section 6, Demonstration Reduction (Data File Entry), above. The command sequence is repeated below, without explanation.

ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).

1 DOMINANT ROOT SELECTION

2 RESIDUE RETENTION

3 DIFFERENTIATION OF POLYNOMIALS

4 "MOORE" REDUCTION - CONTINUOUS

5 "MOORE" REDUCTION - DISCRETE

6 SYSTEM MODELS INDEX

7 LOAD A SYSTEM MODEL

ENTER SELECTED OPTION NUMBER 7

THIS BRANCH OF "MODRED" LOADS A SYSTEM MODEL. THE MODEL
IS LOADED ONTO THE STACK AND INTO A FILE NAMED 'SYSB'.
'SYSB' IS USED BY METHODS USING STATE-VARIABLE MODELS.
THE STATE-VARIABLE MODEL SHOULD DEFINE F,G,H,J, AND TS
IN THE USUAL STATE-VARIABLE MODEL (TS IS THE SAMPLING
INTERVAL, WHERE APPLICABLE).

* * * * * WAITING. CR TO CONTINUE. * * * * *

PAUSE

THE STACK IS USED BY SOME SISO (TRANSFER FUNCTION) METHODS.
IF SUCH APPLICATIONS ARE PLANNED, THE DATA FILE SHOULD
ALSO CONTAIN A VECTOR 'BI' CONTAINING THE
COEFFICIENTS OF THE NUMERATOR POLYNOMIAL, AND A VECTOR
'AI' CONTAINING THE COEFFICIENTS OF THE DENOMINATOR
POLYNOMIAL. THE DENOMINATOR POLYNOMIAL SHOULD BE
NORMALIZED WITH A LEADING COEFFICIENT (HIGHEST POWER OF S)
OF ONE. THE ONE IS NOT INCLUDED IN 'AI".
ENTER THE NAME OF THE FILE CONTAINING THE SYSTEM MODEL.
THE FILENAME MUST VE ENCLOSED IN APOSTROPHES.
E.G. 'FILENAME'

```
ENTER FILENAME 'SYSX'

   --  LOAD  COMPLETED  --

     ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).
          1 DOMINANT ROOT SELECTION
          2 RESIDUE RETENTION
          3 DIFFERENTIATION OF POLYNOMIALS
          4 "MOORE" REDUCTION - CONTINUOUS
          5 "MOORE" REDUCTION - DISCRETE
          6 SYSTEM MODELS INDEX
          7 LOAD A SYSTEM MODEL
     ENTER SELECTED OPTION NUMBER
```

9. EVALUATING RESULTS:

The user is provided two methods for evaluating the success of the reduction procedure. The first (and probably most generally useful) is a plot (or set of plots) providing visual comparison of the response of the full system with that of the reduced system. These plots also show the error function representing the instantaneous difference between the two. The User should interpret both the magnitude and the shape of the response and error function in the context of the particular application.

MATRIX$_x$ scales the plots, automatically, so that the largest value of any of the functions will fill the ordinate of the plot. In the case of an unstable system, there will be exponential growth of the response due to the unstable eigenvalue(s). This can lead to very large values which dominate the selection of the scale factor. The time displayed on the plot should be adjusted (by appropriate selection of the sampling time and number of points chosen for simulation during the reduction process) so that this does not obscure significant transients near the origin.

The other tool for evaluation is the RMS error displayed near the end of the reduction process. This is a quantitative measure that has been provided to evaluate and compare the performance of the various algorithms as well as to evaluate and compare the performance of different levels of reduction of a single system.

The formula operates on the data produced during simulation, so it is a sampled-data approximation of the true RMS error. The formula implemented is:

$$\text{ERROR} = \frac{1.0}{nm}\left\{\sum_{k=1}^{n}\sum_{j=1}^{m}\left[\frac{yf(j,k)-yr(j,k)}{yf(j,k)}\right]^2\right\}^{\frac{1}{2}}$$

In this formula, n is the number of points in each response simulation, m is the number of system responses (i.e. the number of input variables times the number of output variables), yf(j,k) is the response of the full system, and yr(j,k) is the response of the reduced system.

10. DOMINANT ROOT SELECTION:

Dominant Root Selection is the classical method of time-domain model reduction of single-input single-output systems presented in introductory texts such as"Modern Control Systems", by R. C. Dorf (ref. 1, pg. 184). In this method the residues of the response are observed and the smaller ones discarded until the remaining roots represent the desired model order (example #1 below). The method does not work well for responses with nearly equal residues (example #2 below). It requires that complex roots (if present) be selected of discarded in pairs (to avoid complex coefficients in the reduced transfer function). It is, however, the easiest to understand and is included in MODRED primarily for tutorial purposes. While the method could be applied to a variety of response functions, the program module RTSEL operates on the step response of the system and the examples below demonstrate the method with step responses.

Example #1:

Given the transfer function: $T(s) = \frac{0.48(S+5)^2}{(S+1)(S+2)(S+6)}$

(Note: This is demonstration system model #1. It is also discussed in examples #3, #5, and #8)

The step response is: $y_u(t) = 1-1.54\epsilon^{-t}+0.54\epsilon^{-2t}-0.004\epsilon^{-6t}$

It is apparent that the response (fig. 2) is hardly changed if the last term is omitted. Thus this third-order system can be represented accurately by the second-order system which has the step response:

$$y_{ur}(t) = 1-1.54\epsilon^{-t}+0.54\epsilon^{-2t}$$

The corresponding transfer function can be found easily by Laplace

transforming the step response and then removing the 1/S factor that represents the step input. The resulting transfer function is:

$$T_r(s) = \frac{0.46(S+4.35)}{(S+1)(S+2)}$$

Example #2:

Given the transfer function: $T(s) = \frac{2.5(S+0.75)(S+1.5)}{(S+1)(S+1.25)(S+2.25)}$

(Note: This is demonstration model #2A. It is also discussed in examples #4, #6, and #9.)

The step response is: $y_u(t) = 1 - \epsilon^{-2.25t} - \epsilon^{-1.25t} + \epsilon^{-t}$

In this example it is apparent that each residue contributes equally to the response and the removal of any component makes a substantial change in the response (fig. 4).

These examples show that in some cases, at least, a reduced model will provide essentially the same results as the full model. They also show that a reduction method which is effective in one situation may not be effective in every case. Dominant Root Selection is generally effective when some of the poles in the transfer function are closely paired with zeroes, or have negative real components which are much larger than others.

11. RESIDUE RETENTION:

Residue Retention is an exrapolation of the dominant root method discussed above. Residue Retention is presented by Walker and Gupta (ref.4). In this method a constant is added to the response to compensate for the residues which are being discarded. Usually, the constant is the sum of the "DC Gains" of the residues (obtained by evaluating the Laplace transform of the residues at s=0). Equally appropriate constants may be obtained by evaluating the Laplace transform of the residues at some other relevant frequency. The method is usually more accurate than Dominant Root Selection, but is similarly ineffective for responses with nearly equal residues and requires that complex roots be discarded in conjugate pairs. The method can be applied to any response function, the program module RESREN operates on the step response of the system and the examples below illustrate the method with step responses.

Example #3:

Given the transfer function: $T(s) = \frac{0.48(S+5)^2}{(S+1)(S+2)(S+6)}$

(Note: This is system model #1, also discussed in examples #1,

#5, and #8.)

The step response is: $y_u(t) = 1-1.54\epsilon^{-t}+0.54\epsilon^{-2t}-0.004\epsilon^{-6t}$

It is apparent that the response is hardly changed if the last term is omitted. The Laplace transform of this term is:

$$F(S) = \frac{-0.004}{S+6}$$

Evaluation of this expression when s = 0 gives the retention constant -0.0006. Thus this third-order system is represented (fig. 5) by the second-order system which has the step response:

$$y_{ur}(t) = 1-1.54\epsilon^{-t}+0.54\epsilon^{-2t}-0.0006$$

The corresponding transfer function can be found easily by Laplace transforming the step response and then removing the 1/S factor that represents the step input. The resulting transfer function is:

$$T_r(s) = \frac{-0.0007S^2+0.458(S+4.36)}{(S+1)(S+2)}$$

Clearly, these results (fig. 5) are barely distinguishable from the results obtained by Dominant Root Selection above (fig. 2).

Example #4:

Given the transfer function: $T(s) = \frac{2.5(S+0.75)(S+1.5)}{(S+1)(S+1.25)(S+2.25)}$

(Note: This is system model #2A, also discussed in examples #2, #6, and #9.)

The step response is: $y_u(t) = 1 - \epsilon^{-2.25t} - \epsilon^{-1.25t} + \epsilon^{-t}$

In this example it is apparent that each residue contributes equally to the response and the removal of any component makes a substantial change in the response. For this illustration, the root at -2.25 is discarded. The Laplace transform of the discarded residue is:

$$F(s) = \frac{-1}{S+2.5}$$

Evaluation of this expression when s = 0 gives the retention constant -0.4. Thus this third-order system is represented (fig. 6) by the second-order system which has the step response:

$$y_{ur}(t) = 0.96 - \epsilon^{-1.25t} + \epsilon^{-t}$$

The corresponding transfer function is:

$$T_r(s) = \frac{0.96(S+0.72)(S+1.25)}{(s+1)(S+1.25)}$$

In this example the residue retention constant has had little effect on the initial error of the response and has introduced a steady-state error which was not present in the Dominant Root reduction.

Residue Retention is generally comparable to Dominant Root Selection and the modest benefits obtained seldom warrant the added complexity.

12. DIFFERENTIATION OF POLYNOMIALS:

Differentiation of Polynomials uses differentiation to reduce the order of the numerator and denominator polynomials of a Laplace function. The method is presented by Gutman, Mannerfelt, and Molander (ref. 5). Their presentation permits selected poles and zeroes to be excluded from the reduction procedure and for the gain of the reduced model to be adjusted to match the gain of the full model at some selected frequency. These two features are not included in MODRED, but may be incorporated easily in the associated pencil-and-paper analyses. The method provides reliable reduction with moderate accuracy, producing neither high accuracy nor large error, and can produce a real-pole reduction of a complex pair when needed to obtain a reduction of desired order. The Laplace function used for reduction could be either a response fuction or a transfer function. The program module DIFPN assumes that a transfer function is supplied.

Given a polynomial:

$$p_n(s) = a_n s^n + a_{n-1}s^{n-1} + \ldots + a_1 s + a_0$$

The poloynomial is reduced one order by the formula:

$$p_{n-1}(s) = p_n(s) - \frac{s}{n}\left(\frac{dp_n(s)}{ds}\right)$$

Since the first term of the derivative is always $\frac{n}{s}$ times the first term of the polynomial, the formula assures at least a one-order reduction in the polynomial. The algorithm is applied to both the numerator and denominator polynomials. The algorithm is repeated for additional order reduction.

Example #5:

Given the transfer function: $T(s) = \frac{0.48(S+5)^2}{(S+1)(S+2)(S+6)} = \frac{0.48S^2+4.8S+12}{S^3+9S^2+20S+12}$

(Note: This is system model #1, also discussed in examples #1, #3, and #8.)

The step response is: $y_u(t) = 1-1.54\epsilon^{-t}+0.54\epsilon^{-2t}-0.004\epsilon^{-6t}$

The reduced numerator polynomial $q_r(s)$ is obtained from the numerator polynomial $q(s) = 0.48S^2+4.8S+12$:

$$q_r(s) = (0.48S^2+4.8S+12) - \frac{S}{2}(0.96S+4.8) = 2.4S+12$$

Similarly, the reduced denominator polynomial $p_r(s)$ is obtained from the denominator polynomial $p(s) = S^3+9S+20S+12$:

$$p_r(s) = (S^3+9S^2+20S+12) - \frac{S}{3}(3S^2+18S+20) = 3S^2+13.2S+12$$

The resulting transfer function is:

$$T_r(s) = \frac{0.8(S+5)}{S^2+4.4S+4} = \frac{0.8(S+5)}{(S+1.28)(S+3.12)}$$

This produces a step response of:

$$y_{ur}(t) = 1-1.26\epsilon^{-1.28t}+0.26\epsilon^{-3.12t}$$

These results (fig. 7) are not as accurate as the results obtained by Dominant Root Selection above, but do preserve the steady-state value and the overall shape of the response.

Example #6:

Given the transfer function: $T(s) = \frac{2.5(S+0.75)(S+1.5)}{(S+1)(S+1.25)(S+2.25)}$

(Note: This is system model #2A, also discussed in examples #2, #4, and #9.)

The step response is: $y_u(t) = 1 - \epsilon^{-2.25t} - \epsilon^{-1.25t} + \epsilon^{-t}$

In this example it is apparent that each residue contributes equally to the response.

The reduced numerator polynomial is:

$$q_r(s) = (2.5S^2+5.62S+2.81) - \frac{S}{2}(5S+5.62) = 2.81(S+1)$$

The reduced denominator polynomial is:

$$p_r(s) = (S^3+4.5S^2+6.31S+2.81) - \frac{S}{3}(3S^2+9S+6.31) = 1.5(S+1.3)(S+1.5)$$

The step response of the reduced transfer function is:

$$y_{ur}(t) = 1+2.14e^{-1.3t}-3.14e^{-1.5t}$$

This response (fig. 8) is much better than the corresponding results obtained by either dominant root selection (example #2) or residue retention (example #4) above. It is correct for both the initial and final values and has an RMS error of 2% as compared to 8% for the previous methods.

Example #7:

Given the Transfer Function:

$$T(S) = \frac{1.11(S+0.5)^2}{(S+1)(S+0.5-j0.167)(S+0.5+j0.167)} = \frac{1.11S^2+1.11S+0.28}{S^3+2S^2+1.28S+0.28}$$

(Note: This is system model #3B, also discussed in example #10.)

The step response is: $y_u(t) = 1-e^{-t}+0.66e^{-0.5t}\sin 0.167t$

Dominant root selection in this example would require removing one complex root (not possible) to achieve a second-order model. In contrast, differentiation of polynomials can always make a one-order reduction regardless of the type of roots.

The reduced numerator polynomial is:

$$q_r(t) = (1.11S^2+1.11S+0.28) - \frac{S}{2}(2.22S+1.11) = 0.56(S+0.5)$$

The reduced denominator polynomial is:

$$p_r(t) = (S^3+2S^2+1.28S+0.28) - \frac{S}{3}(3S^2+4S+1.28) = 0.67(S+0.64-j0.1)(S+0.63+j0.1)$$

The step response of the reduced transfer function is:

$$y_{ur}(t) = 1+2.33e^{-0.64t}\cos(0.1t-2.04)$$

This response (fig. 9) is better than either the second-order model obtained by discarding the residue of the dominant real eigenvalue or the first-order model obtained by discarding the residues of the non-dominant complex pair using either of the previous methods.

Differentiation of Polynomials seems to provide reliable model with moderate accuracy for any order of reduction and for any type of eigenvalue and residue combination. The simplicity of the mathematical theory is attractive.

13. Subsystem Elimination:

Subsystem elimination is a mathematically complicated method of model reduction based on Principal Component Analysis, Singular Value Decomposition, the Grammian, and the Second-Order Modes of the system. The background for this method is presented by Moore (ref. 6). A detailed discussion is beyond the scope of this user's guide. Briefly, the reduction process subjects the system model to two successive similarity transformations which balance the effects of the inputs on the state variables and the effects of the state-variables on the outputs. The process then computes a set of "condition numbers" which indicate the importance of the states in much the way that residues indicate the importance of the eigenvalues in dominant root reduction (Sec. 10 above). The method has provided accurate results for every system tested, usually surpassing the other methods discussed above by several orders of magnitude.

In "MODRED", balancing and component value computation is performed on the unaugmented state-variable model. Thus, the reduction is based upon the impulse response. Nevertheless, evaluation is based on the step response as for the other methods. Subsystem elimination is applied to continuous state-variable models in the program module "MOORE" and to discrete state-variable models in the program module "DISCRT". Both program modules accomodate stable or unstable models of either scalar or multivariable systems. The algorithm for stable systems in the continuous version is a direct implementation of the algorithm presented by Integrated Systems Incorporated (ref. 3). The algorithm for unstable continuous systems is an original implementation of theory presented by Moore (ref. 6). The algorithm for discrete-time systems is an original adaptation of the unstable continuous algorithm to the discrete-time case. The examples below illustrate discrete-time applications though the discussion and results are generally similar to continuous-time applications as well. The models and results shown are for the sampling interval and number of sample points indicated. Different choices could lead to significantly different results. It may be possible to get even better results and it is generally possible to select choices which will give irrelevant results (see discussion in section 6 above).

Example #8:

Given the Discrete State-Variable Model (T_s=0.1667, Npts=20):

$$\underline{x(k+1)} = \begin{bmatrix} 0.1149 & 0.0786 & 0.0086 \\ -1.6748 & 0.8224 & 0.1556 \\ -0.9433 & -0.1027 & 0.9935 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.0853 \\ 0.8225 \\ 1.9163 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = [1.0\ 0.0\ 0.0]\underline{x(k)} + [0.0]\underline{u(k)}$$

(Note: This is system model #1, also discussed in examples #1, #3, and #5.)

The balanced model of the system is:

$$\underline{x_b(k+1)} = \begin{bmatrix} 0.9015 & -0.0995 & 0.0059 \\ 0.0995 & 0.6711 & 0.0569 \\ 0.0059 & -0.0569 & 0.3583 \end{bmatrix}\underline{x_b(k)} + \begin{bmatrix} 0.3259 \\ -0.1451 \\ -0.0102 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [0.3259\ 0.1451\ -0.0102]\underline{x_b(k)} + [0.0]\underline{u(k)}$$

This balanced model displays the magnitude-symmetry of the balanced model both in the "F" matrix and between the "G" and "H" vectors.

The corresponding component magnitudes (used as "condition numbers") are 32, 1, and 0.001. These indicate the relative contribution of x_{b1}, x_{b2}, and x_{b3} respectively to the system response. The component magnitudes suggest that a good first order approximation is possible in this example. Nevertheless, a second-order reduction is made below, to provide a direct comparison with the other example reductions of system #1.

The balanced system model is reduced to second-order by discarding the elements of the system associated with the x_{b3} subsystem (i.e. the elements in the third row or the third column of "F", "G", and "H").

The reduced balanced system is:

$$\underline{x_{br}(k+1)} = \begin{bmatrix} 0.9015 & -0.0995 \\ 0.0995 & 0.6711 \end{bmatrix}\underline{x_{br}(k)} + \begin{bmatrix} 0.3259 \\ -0.1451 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [0.3259\ 0.1451]\underline{x_{br}(k)} + [0.0]\underline{u(k)}$$

As in examples #1 and #3, the response of the reduced system (fig3) is indistinguishable from that of the full system. This method does, however, provide an order-of-magnitude reduction in the RMS error (from 0.01% to 0.001%).

Example #9:

Given the Discrete State-Variable Model (T_s=0.25, Npts=14):

$$\underline{x(k+1)} = \begin{bmatrix} 0.2272 & 0.1403 & 0.0215 \\ -0.9462 & 0.8586 & 0.2372 \\ -0.3946 & -0.0605 & 0.9944 \end{bmatrix}\underline{x(k)} + \begin{bmatrix} 0.4774 \\ 1.0660 \\ 0.5195 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [1.0\ 0.0\ 0.0]\underline{x(k)} + [0.0]\underline{u(k)}$$

(Note: This is system model #2A, also discussed in examples #2, #4, and #6.)

The balanced model of the system is:

$$\underline{x_b(k+1)} = \begin{bmatrix} 0.5514 & -0.1011 & 0.0196 \\ 0.1011 & 0.8683 & 0.0636 \\ 0.0196 & -0.0636 & 0.6605 \end{bmatrix}\underline{x_b(k)} + \begin{bmatrix} 0.6927 \\ -0.0509 \\ -0.0111 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [0.6927\ 0.0509\ -0.0111]\underline{x_b(k)} + [0.0]\underline{u(k)}$$

This balanced model also displays the magnitude-symmetry of the balanced model both in the "F" matrix and between the "G" and "H" vectors.

The corresponding component magnitudes (used as "condition numbers") are 23, 0.14, and 0.00043. These indicate the relative contribution of x_{b1}, x_{b2}, and x_{b3} respectively to the system response. The component magnitudes suggest that x_{b3} contributes little to the system response. While x_{b2} contributes much more, it too may well be insignificant in comparison to x_{b1}. Nevertheless, a second-order reduction is made below, to provide a direct comparison with the other example reductions of system #2A.

The balanced system model is reduced to second-order by discarding the elements of the system associated with the x_{b3} subsystem (i.e. the elements in the third row or the third column of "F", "G", and "H").

The reduced balanced system is:

$$\underline{x_{br}}(k+1) = \begin{bmatrix} 0.5514 & -0.1011 \\ 0.1011 & 0.8683 \end{bmatrix}\underline{x_{br}}(k) + \begin{bmatrix} 0.6927 \\ -0.0509 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [0.6927 \;\; 0.0509]\underline{x_{br}}(k) + [0.0]\underline{u(k)}$$

In contrast to examples #2, #4, and #6 where the other methods failed to make an accurate reduction, the response of this reduced system (fig.10) is indistinguishable from that of the full system. This method reduces the RMS error in the reduced model from 2-8% to 0.007%.

Example #10:

Given the Discrete State-Variable Model (T_s=0.175, Npts=20):

$$\underline{x(k+1)} = \begin{bmatrix} 0.6890 & 0.1467 & 0.0136 \\ -0.1912 & 0.9824 & 0.1739 \\ -0.0407 & -0.0038 & 0.9998 \end{bmatrix}\underline{x(k)} + \begin{bmatrix} 0.1784 \\ 0.1779 \\ 0.0441 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [1.0 \;\; 0.0 \;\; 0.0]\underline{x(k)} + [0.0]\underline{u(k)}$$

(Note: This is system model #3B, also discussed in example #7.)

The balanced model of the system is:

$$\underline{x_b(k+1)} = \begin{bmatrix} 0.8307 & 0.0443 & 0.0074 \\ -0.0443 & 0.9608 & -0.0360 \\ 0.0074 & 0.0360 & 0.8787 \end{bmatrix}\underline{x_b(k)} + \begin{bmatrix} 0.4250 \\ 0.0477 \\ -0.0086 \end{bmatrix}\underline{u(k)}$$

$$\underline{y(k)} = [0.4250 \;\; -0.0477 \;\; -0.0086]\underline{x_b(k)} + [0.0]\underline{u(k)}$$

This balanced model also displays the magnitude-symmetry of the balanced model both in the "F" matrix and between the "G" and "H" vectors.

The corresponding component magnitudes (used as "condition numbers") are 35, 0.06, and 0.00005. These indicate the relative contribution of x_{b1}, x_{b2}, and x_{b3} respectively

to the system response. The component magnitudes suggest that neither x_{b2} nor x_{b3} contribute much to the system response. Nevertheless, a second-order reduction is made below, to provide a direct comparison with the other example reductions and especially the other reduction of system #3B.

The balanced system model is reduced to second-order by discarding the elements of the system associated with the x_{b3} subsystem (i.e. the elements in the third row or the third column of "F", "G", and "H").

The reduced balanced system is:

$$\underline{x_{br}}(k+1) = \begin{bmatrix} 0.8307 & 0.0443 \\ -0.0443 & 0.9608 \end{bmatrix} \underline{x_{br}}(k) + \begin{bmatrix} 0.4250 \\ 0.0477 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = [0.4250 \quad -0.0477] \underline{x_{br}}(k) + [0.0] \underline{u(k)}$$

In contrast to example #7, the response of this reduced system (fig.11) is indistinguishable from that of the full system. This method reduces the RMS error in the reduced model from 2.7% to 0.00003%.

Example #11:

Given the Discrete State-Variable Model (T=0.1667, Npts=21):

$$\underline{x(k+1)} = \begin{bmatrix} 0.1149 & 0.0786 & 0.0086 \\ -1.6749 & 0.8223 & 0.1557 \\ -0.9433 & -0.1027 & 0.9935 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.0853 & 0.4952 \\ 0.8227 & -0.5938 \\ 1.9166 & 1.3677 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = \begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.8946 & 0.1440 & 0.4230 \end{bmatrix} \underline{x(k)} + [0.0] \underline{u(k)}$$

(Note: This is system model #4A.)

The balanced model of the system is:

$$\underline{x_b(k+1)} = \begin{bmatrix} 0.9094 & 0.0935 & -0.1037 \\ -0.0467 & 0.8532 & -0.3064 \\ -0.0689 & 0.3651 & 0.1682 \end{bmatrix} \underline{x_b(k)} + \begin{bmatrix} 0.8461 & 0.6963 \\ -0.2069 & 0.5350 \\ -0.0295 & 0.7674 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = \begin{bmatrix} 0.4250 & -0.0477 & -0.0086 \\ 1.0954 & -0.5035 & 0.5769 \end{bmatrix} \underline{x_b(k)} + [0.0]\underline{u(k)}$$

This is system #1 modified by the addition of a second input and a second output. The balanced model shows the distortion of magnitude-symmetry typical of multivariable models.

The component magnitudes (used as "condition numbers") are 1452, 1167, and 378. These indicate the relative contribution of x_{b1}, x_{b2}, and x_{b3} respectively to the system response. The component magnitudes suggest that all three state variables contribute to the system response. Nevertheless, a second-order reduction is made below, to provide a direct comparison with the other example reductions.

The balanced system model is reduced to second-order by discarding the elements of the system associated with the x_{b3} subsystem (i.e. the elements in the third row or the third column of "F", "G", and "H").

The reduced balanced system is:

$$\underline{x_{br}(k+1)} = \begin{bmatrix} 0.9094 & 0.0935 \\ -0.0467 & 0.8532 \end{bmatrix} \underline{x_{br}(k)} + \begin{bmatrix} 0.8461 & 0.6963 \\ -0.2069 & 0.5350 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = \begin{bmatrix} 0.4250 & -0.0477 \\ 1.0954 & -0.5035 \end{bmatrix} \underline{x_{br}(k)} + [0.0]\underline{u(k)}$$

This multivariable system has four step responses (figs. 12 through 15) and the algorithm must try to reflect all four in the reduced model. The results are not as good as for the single-input single-output systems above (examples #8 through #10). Nevertheless, the RMS error (0.4%) obtained under these difficult conditions are nearly as good as the best results

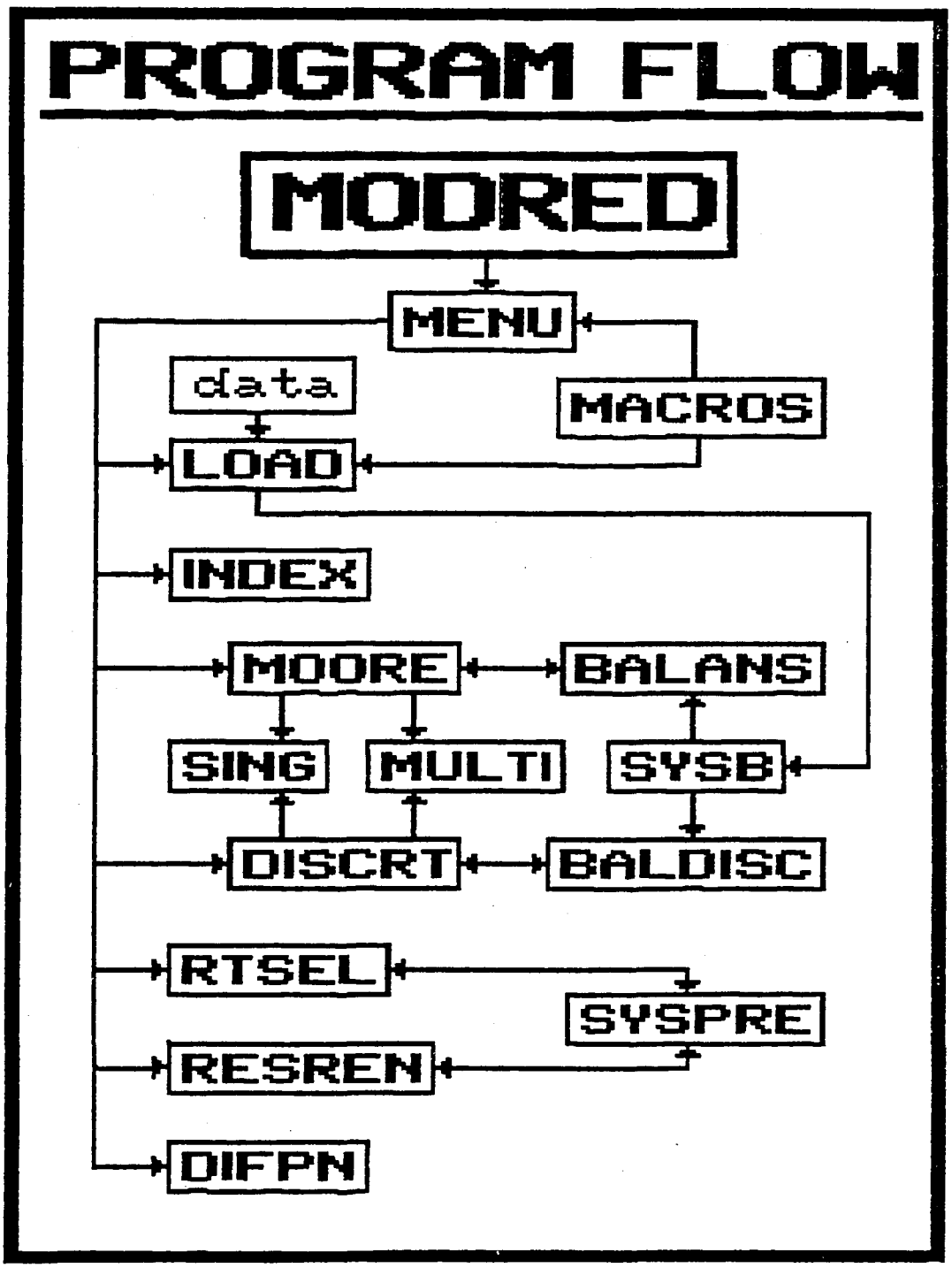
acheived by the other methods under the most favorable conditions. In viewing the response graphs, note that the scales are not uniform, thus making the errors look inconsistent among the various graphs.

* * * END * * *

MODRED USER'S GUIDE

Appendix A

Figures


PROGRAM FLOW
MODRED
MENU
data
MACROS
LOAD
INDEX
MOORE
BALANS
SING
MULTI
SYSB
DISCRT
BALDISC
RTSEL
SYSPRE
RESREN
DIFPN

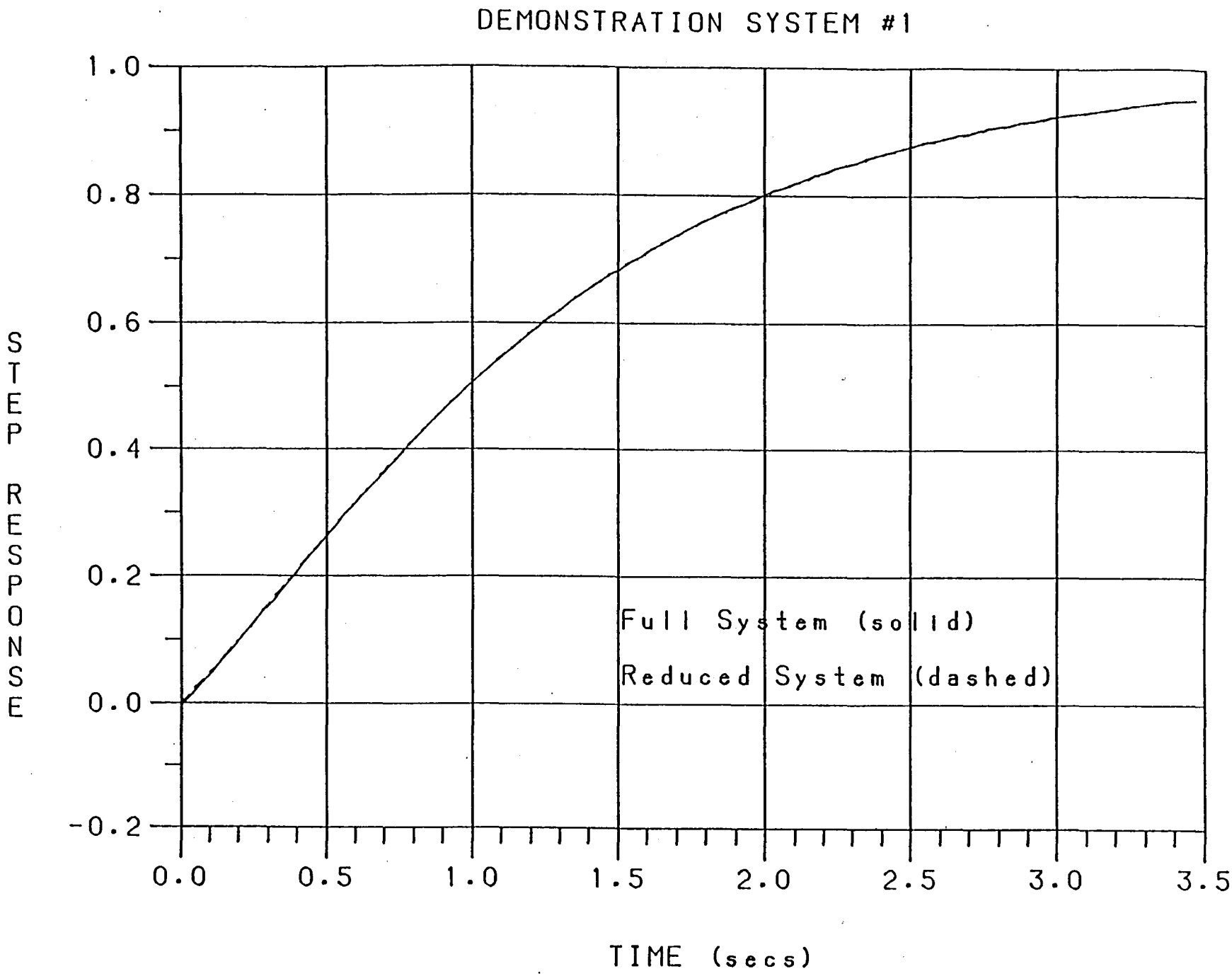


FIGURE 2 - DOMINANT ROOT SELECTION

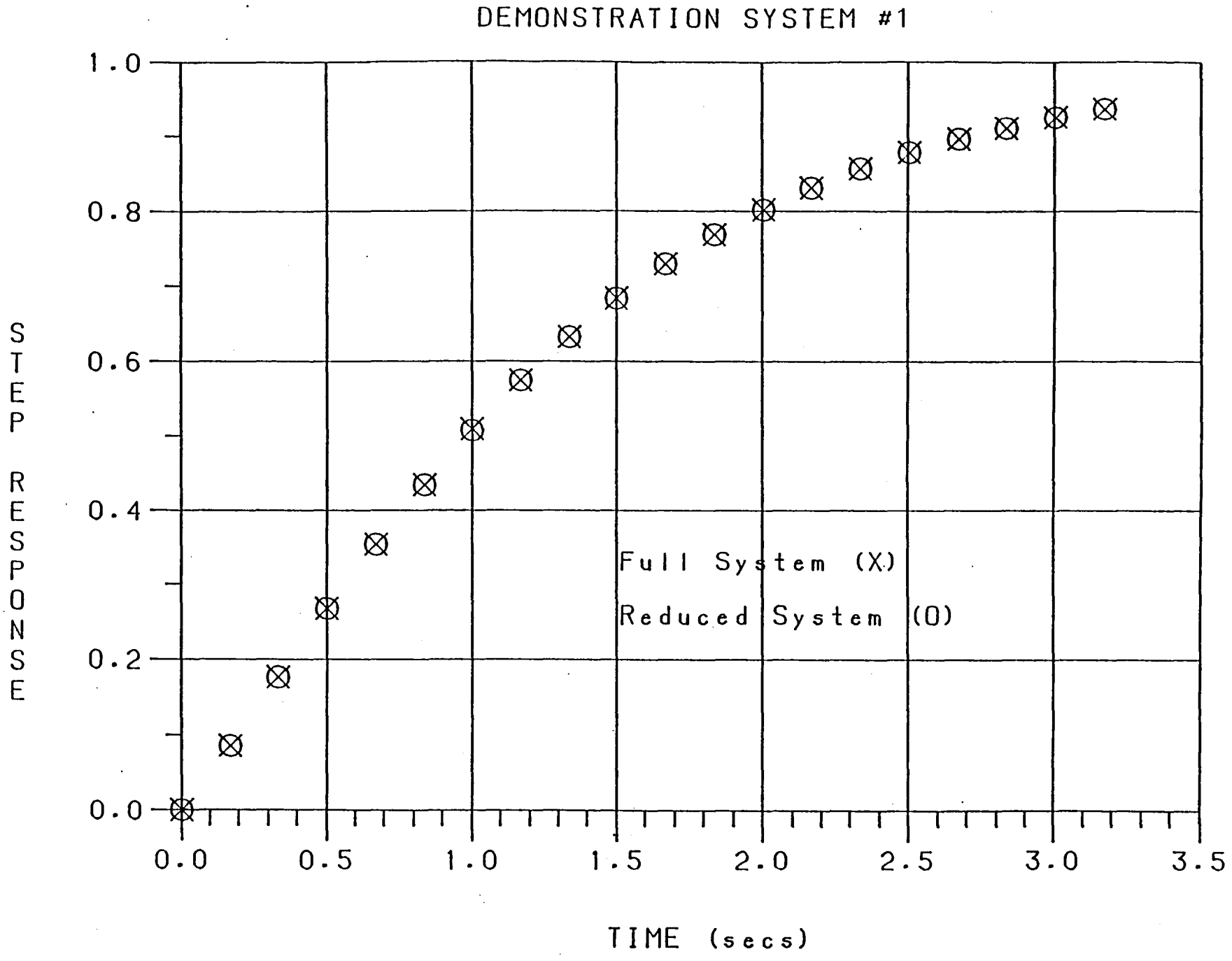


FIGURE 3 SUBSYSTEM ELIMINATION

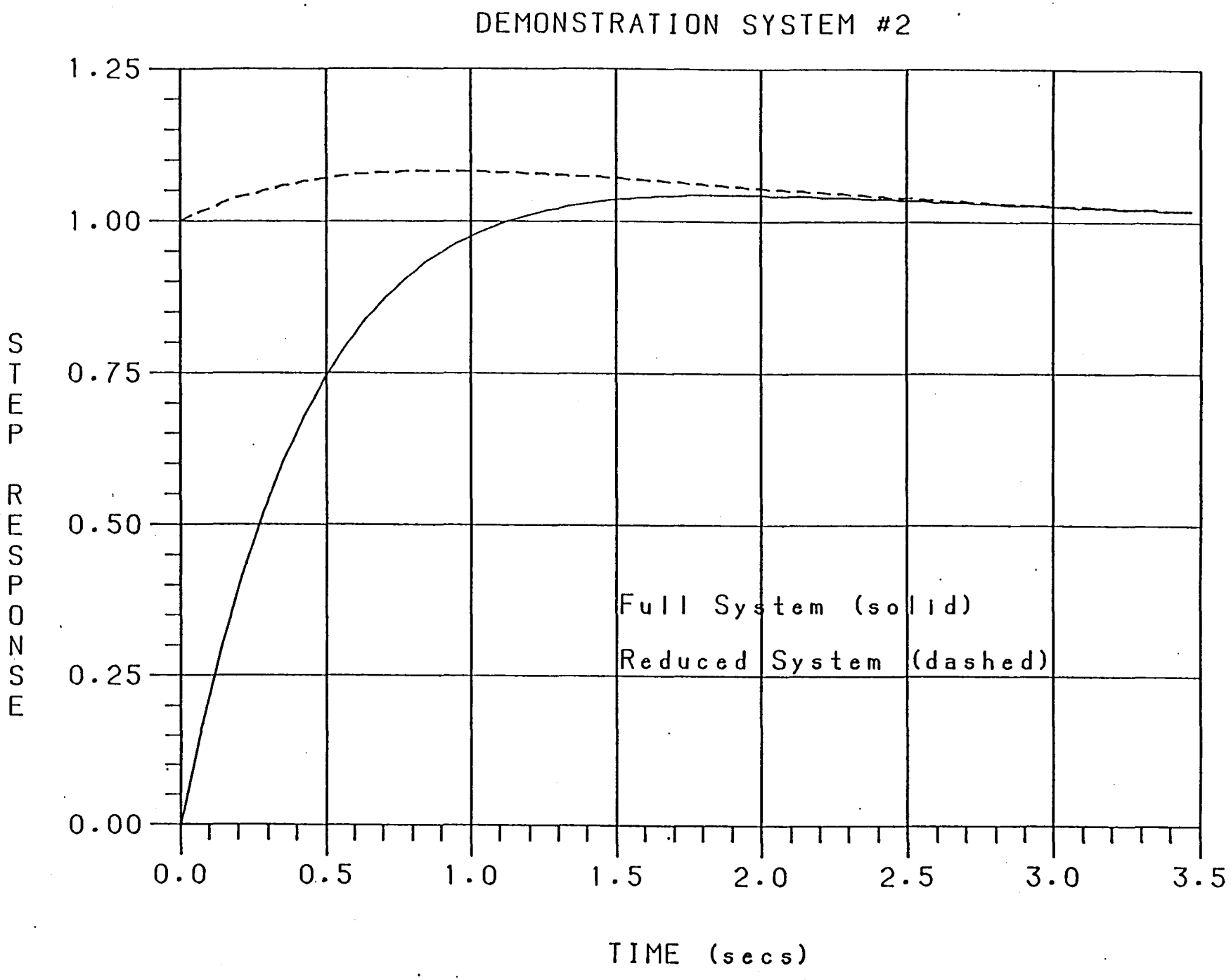


FIGURE 4 DOMINANT ROOT SELECTION

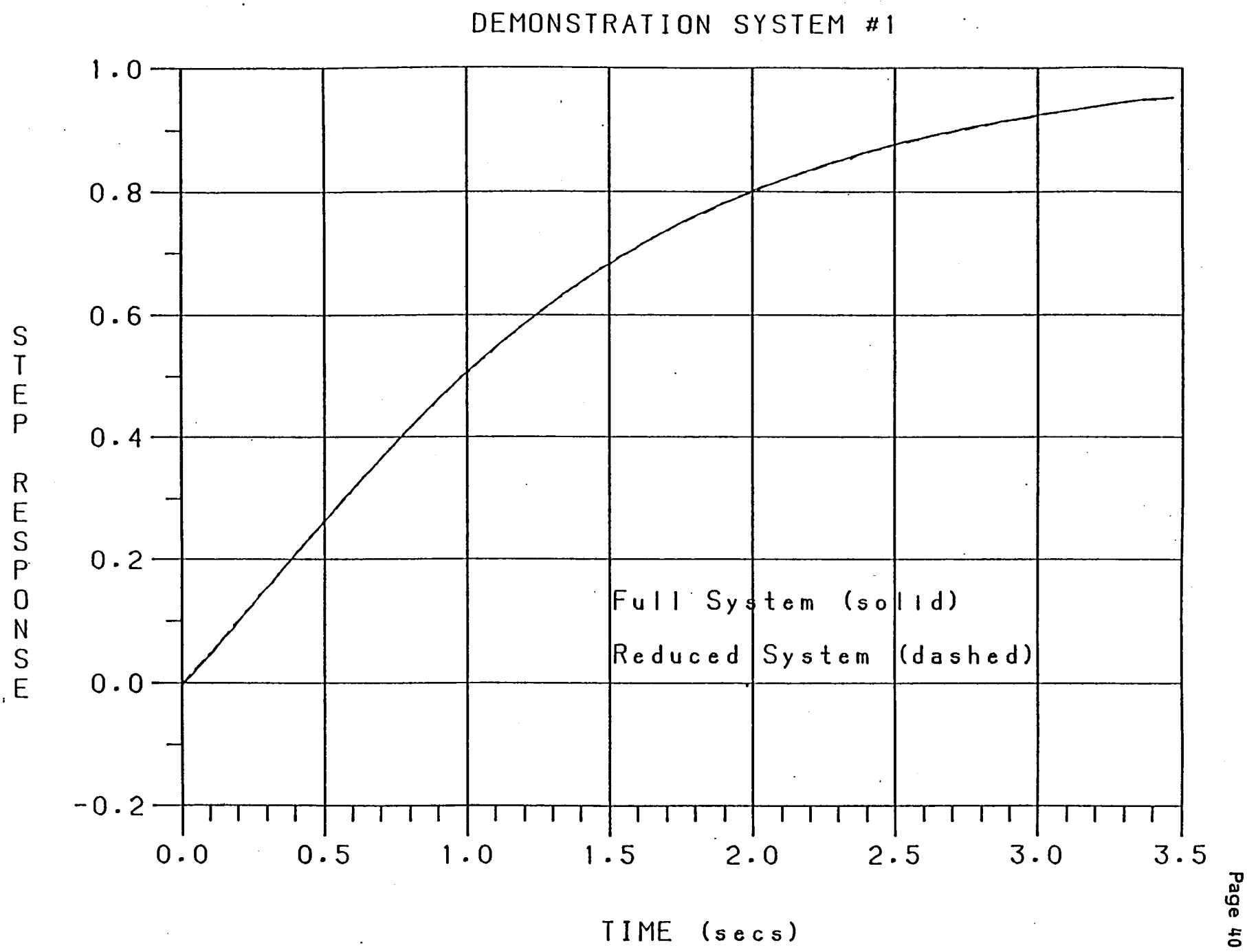


FIGURE 5 RESIDUE RETENTION

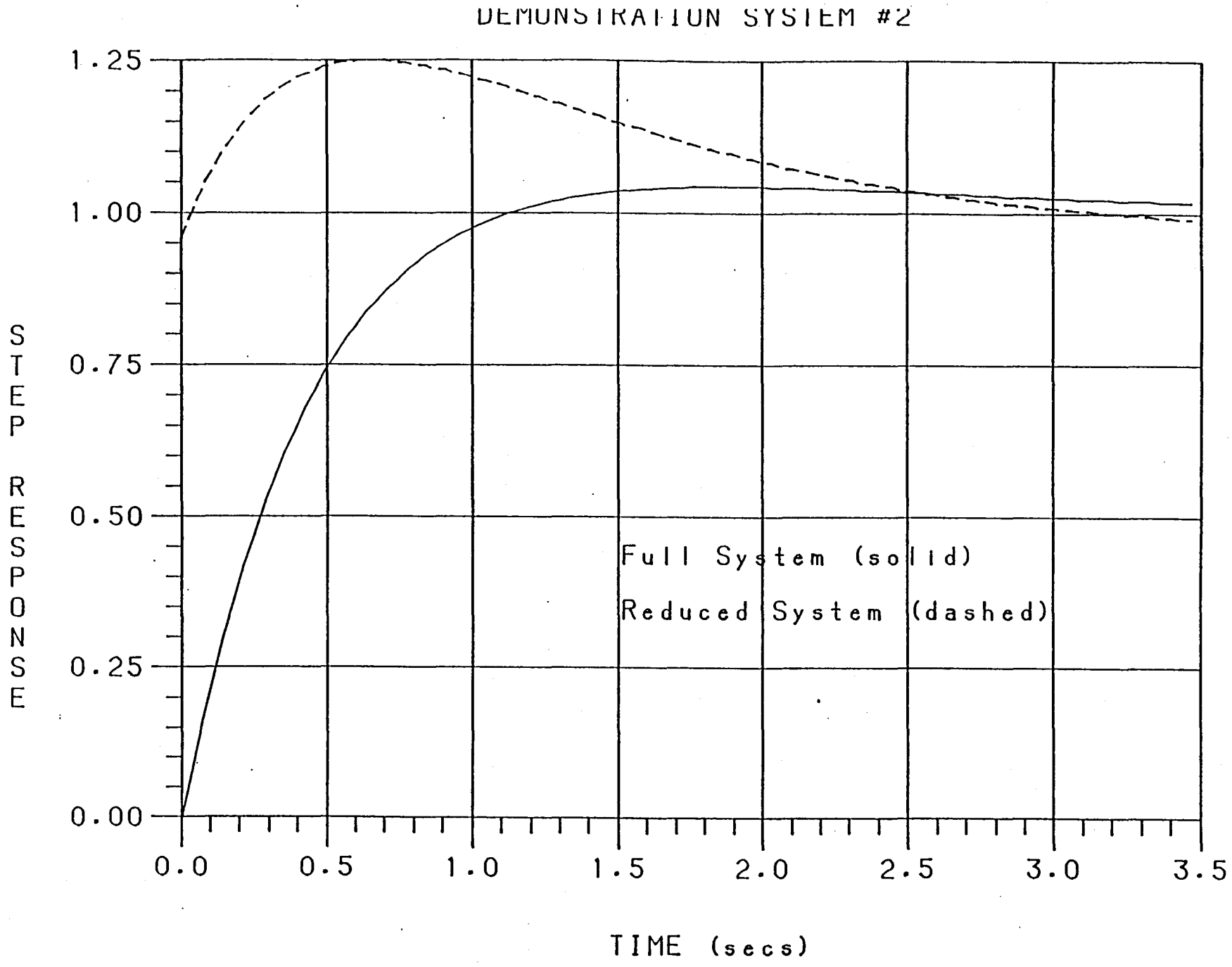


FIGURE 6 RESIDUE RETENTION

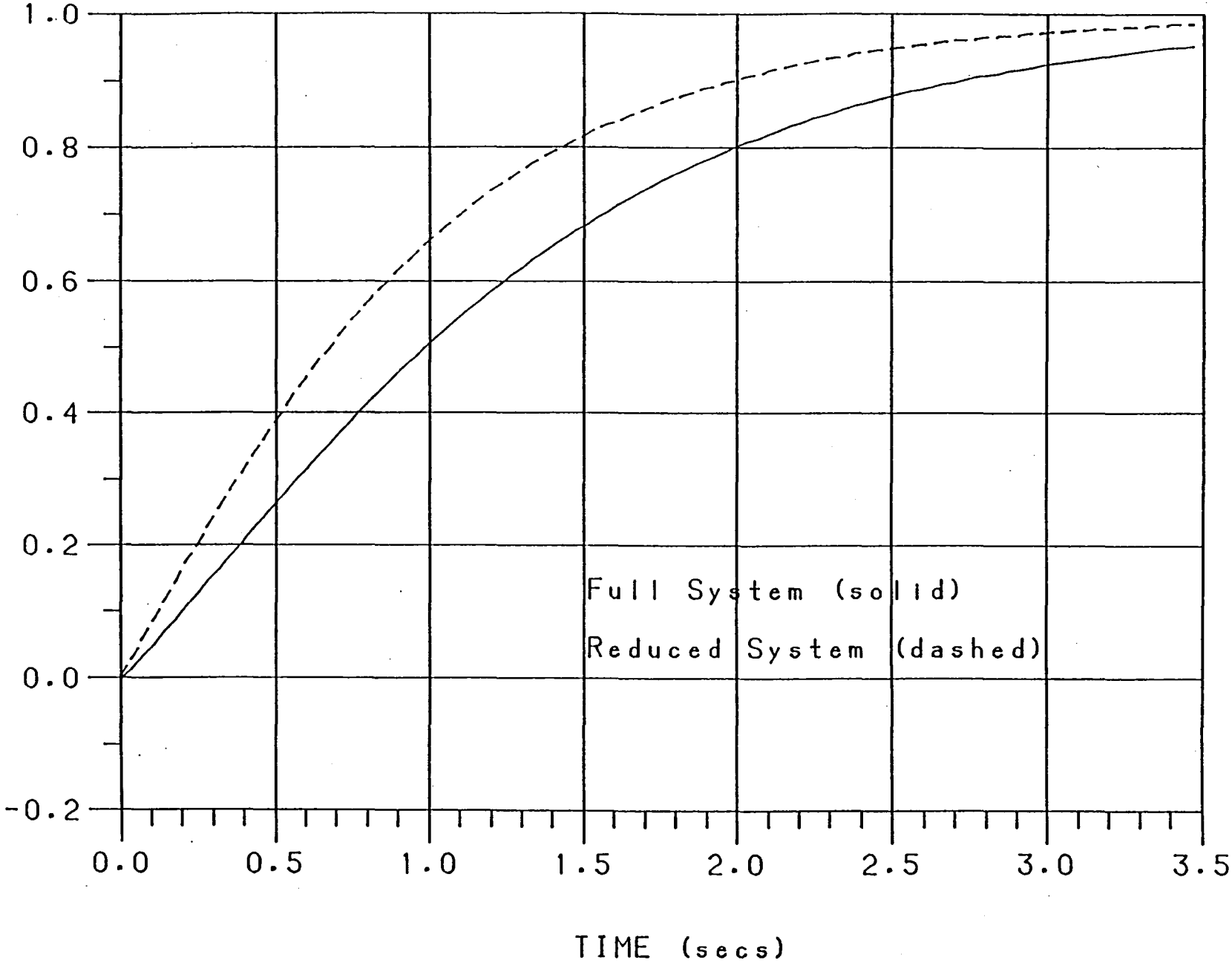


FIGURE 7 - DIFFERENTIATION OF POLYNOMIALS

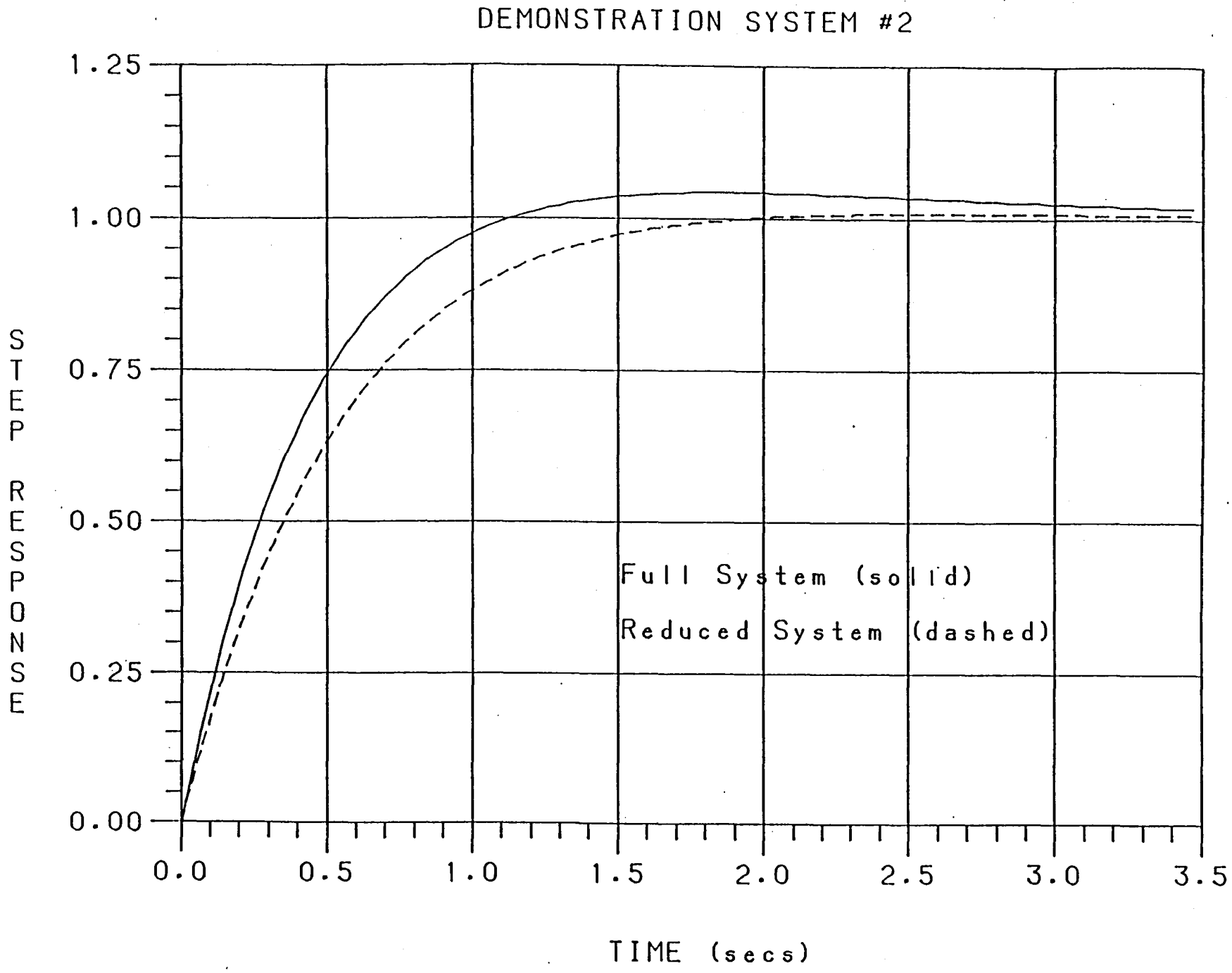


FIGURE 8 - DIFFERENTIATION OF POLYNOMIALS

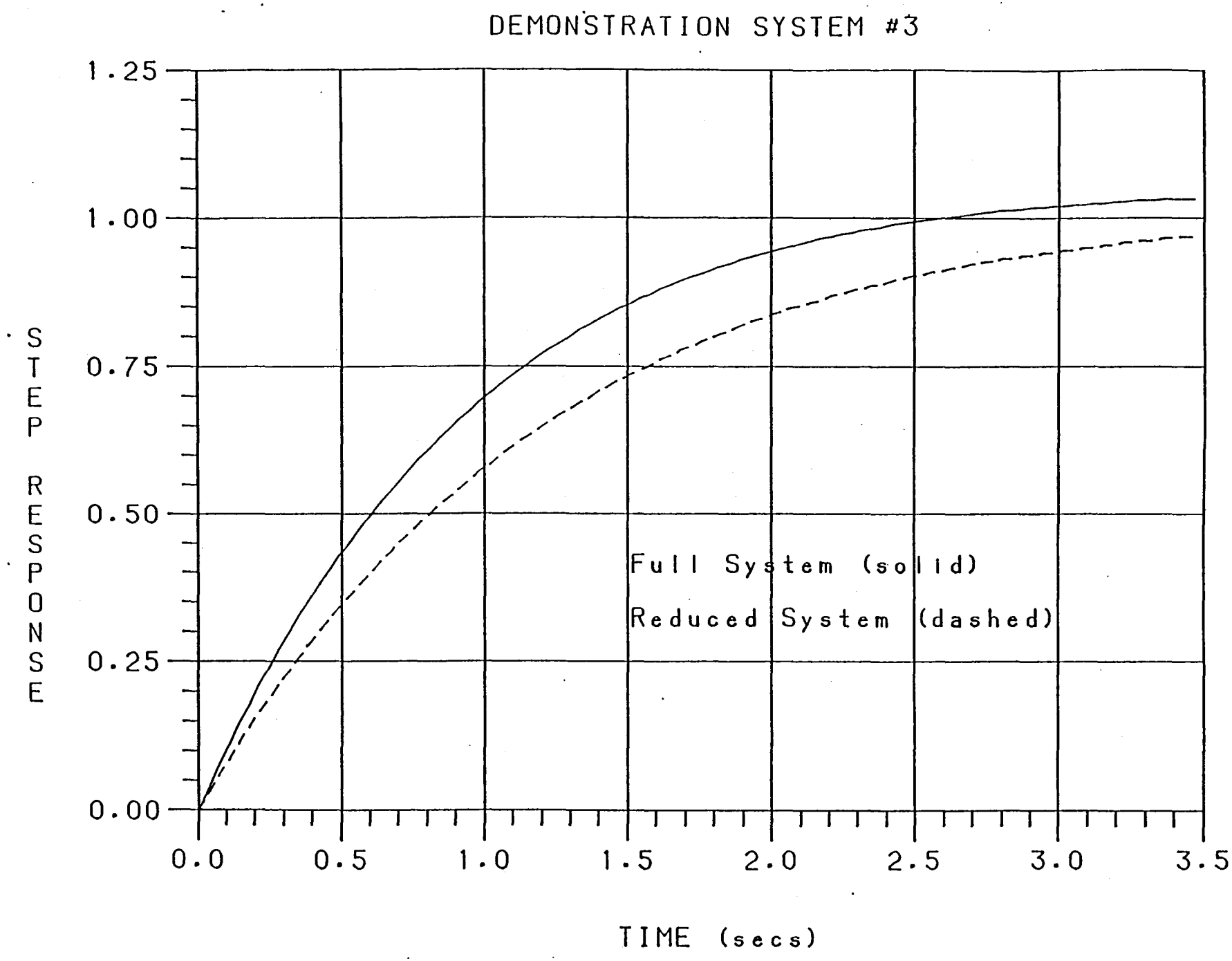


FIGURE 9 - DIFFERENTIATION OF POLYNOMIALS

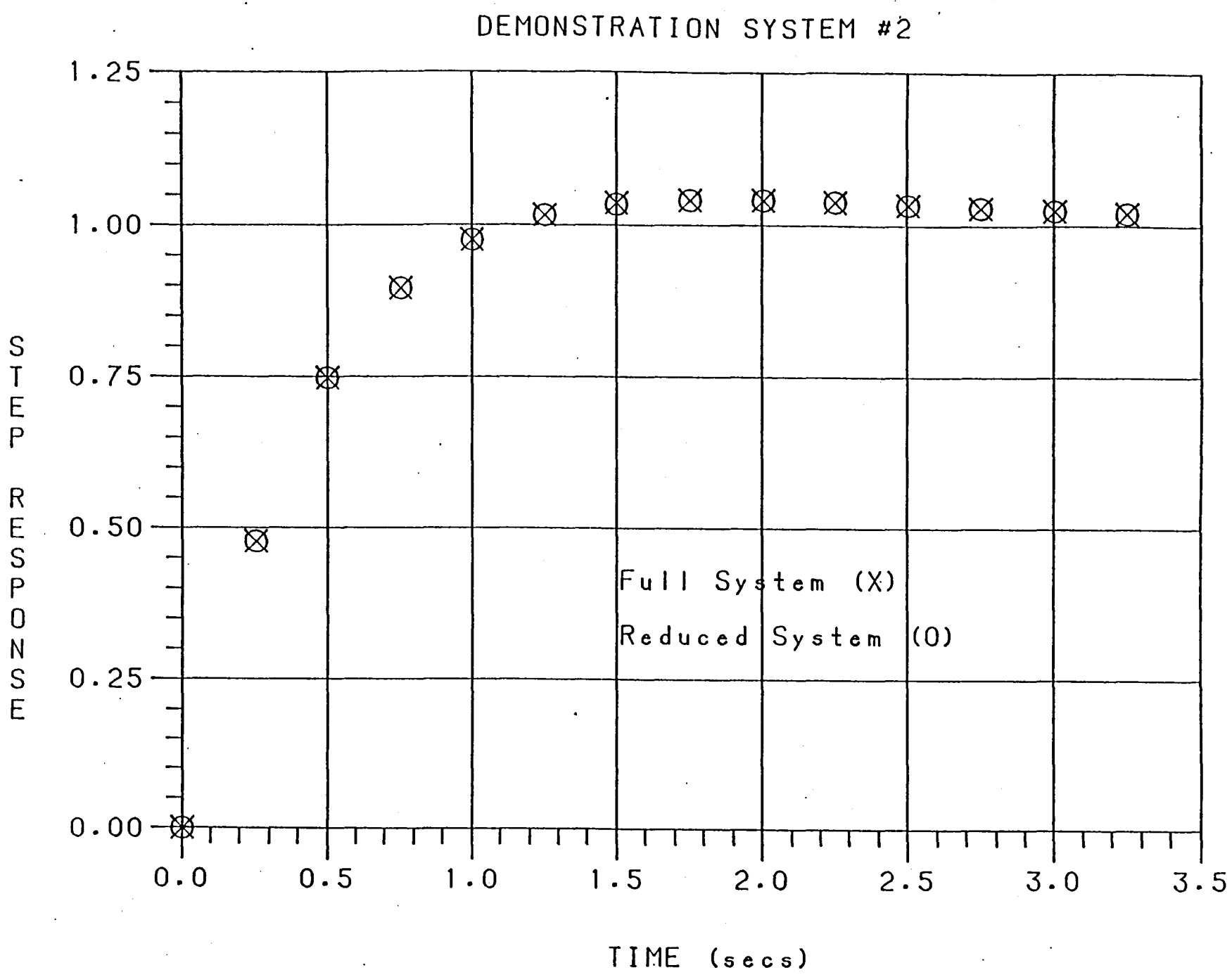


FIGURE 10 SUBSYSTEM ELIMINATION

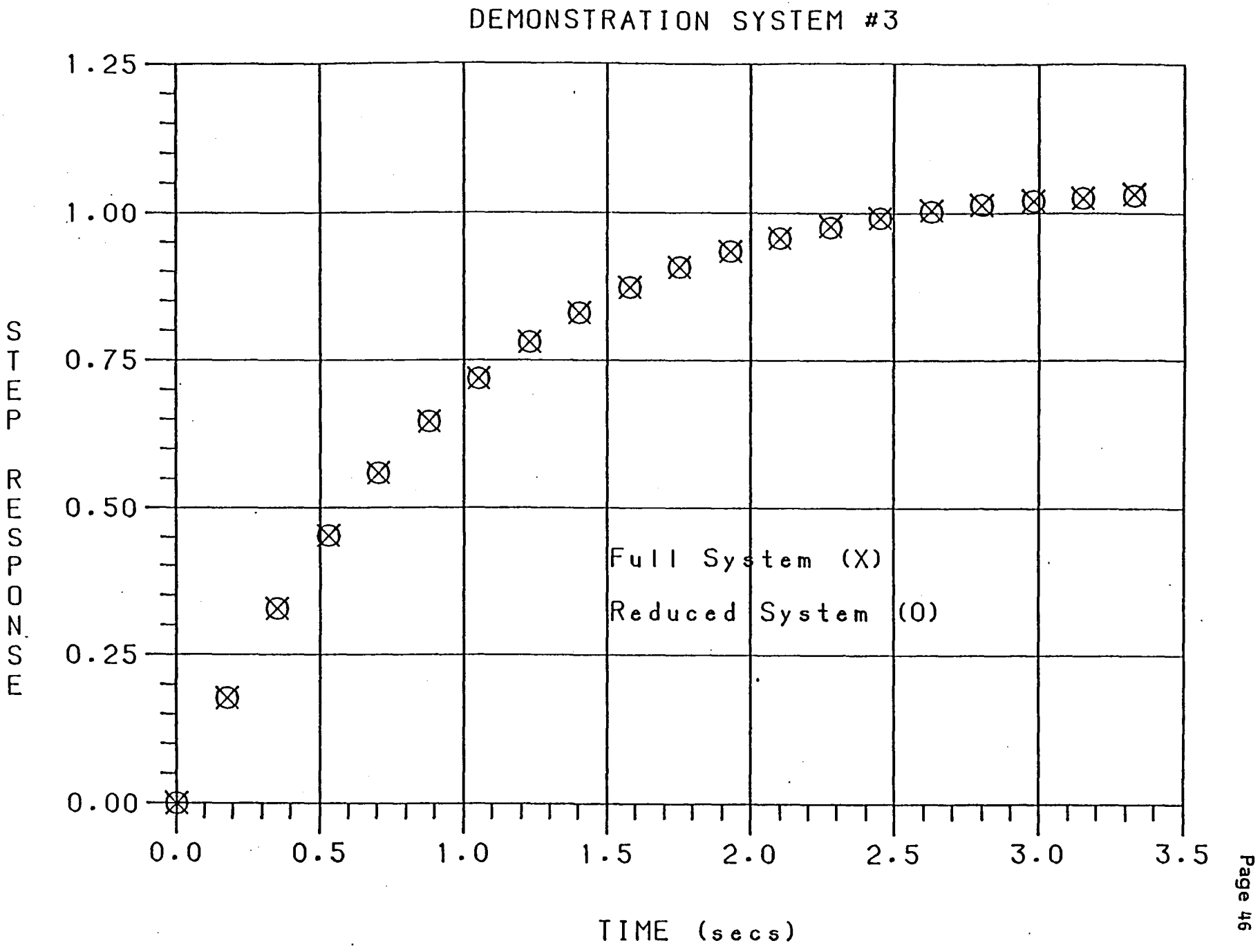


FIGURE 11 SUBSYSTEM ELIMINATION

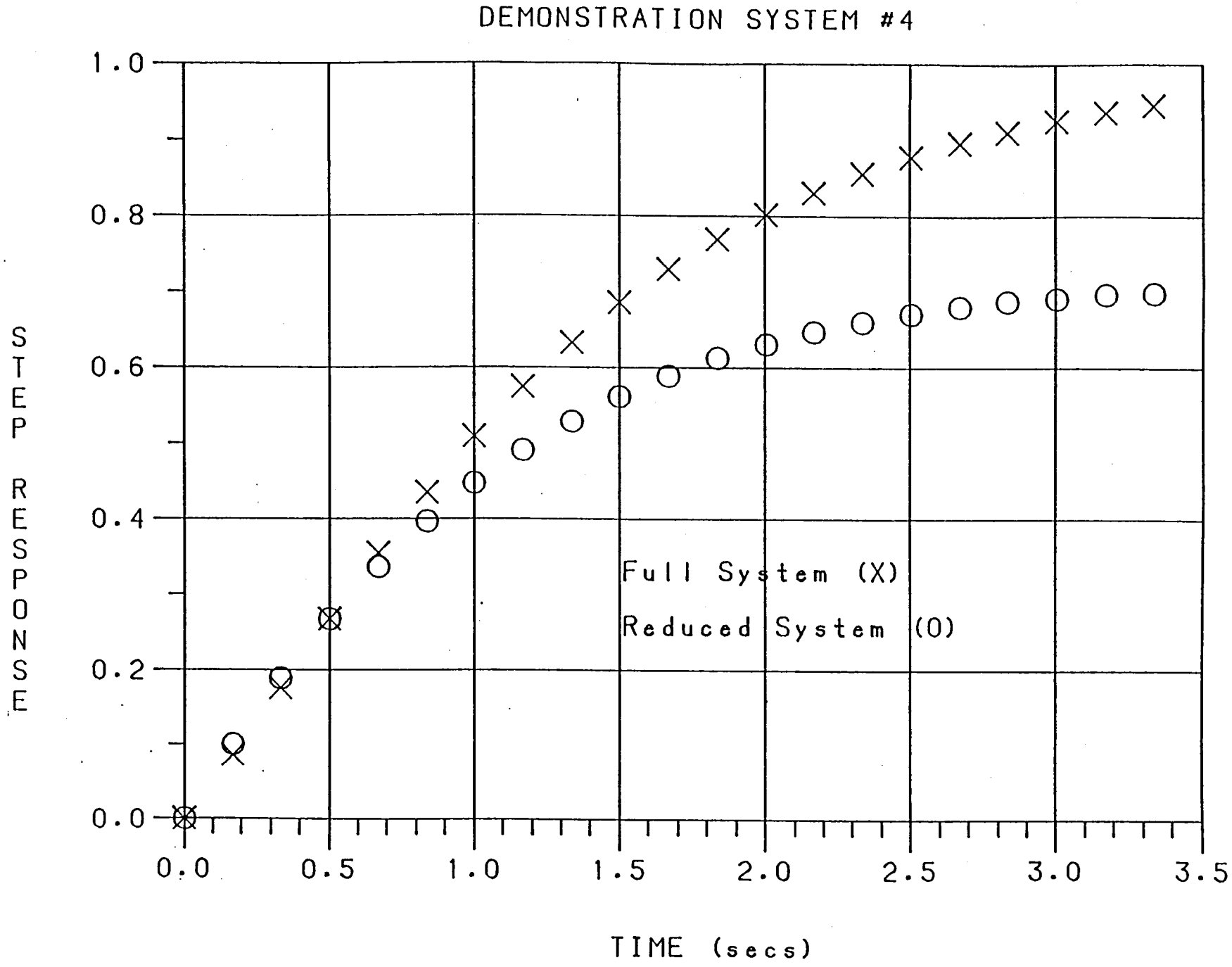


FIGURE 12 SUBSYSTEM ELIMINATION

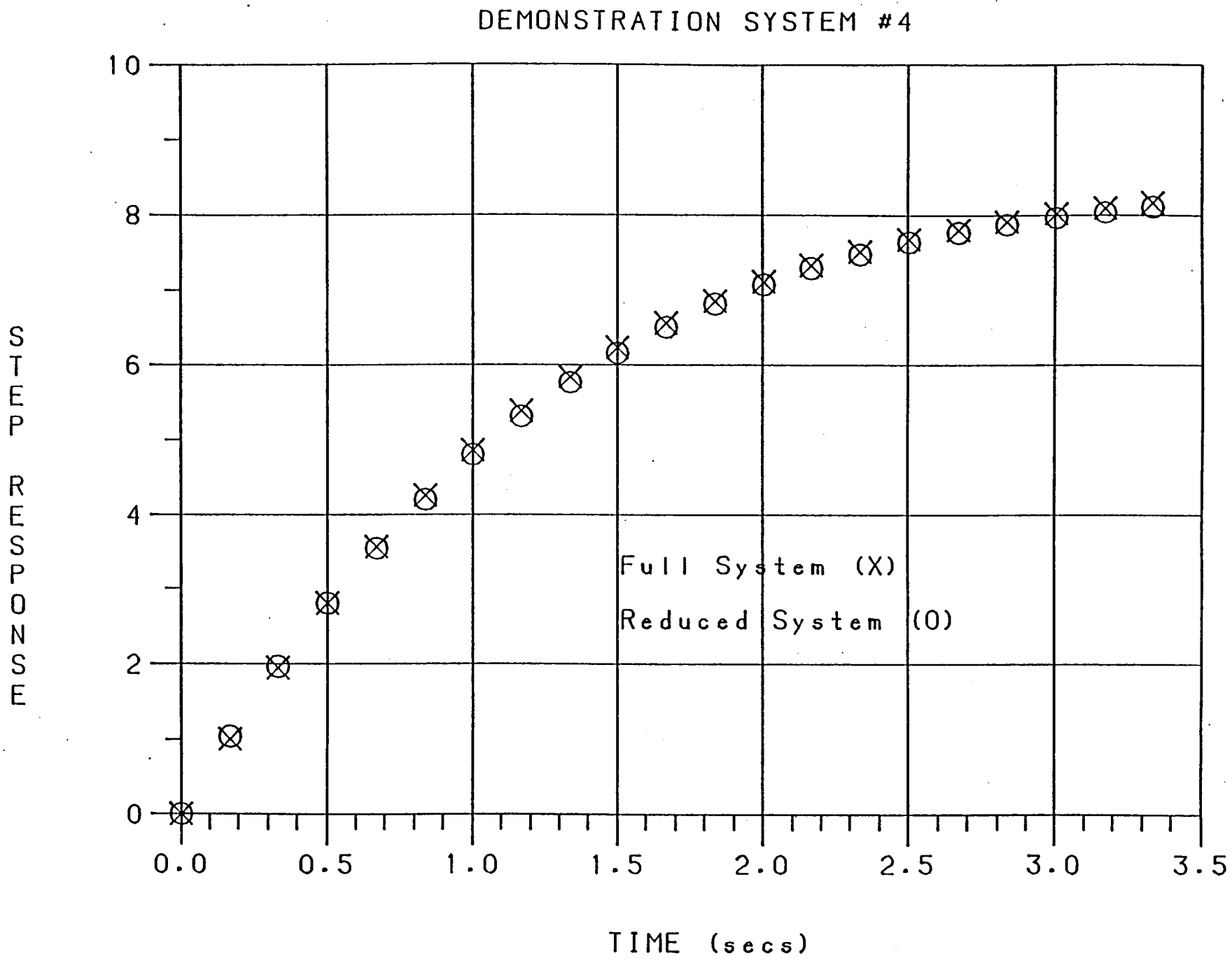


FIGURE 13 SUBSYSTEM ELIMINATION

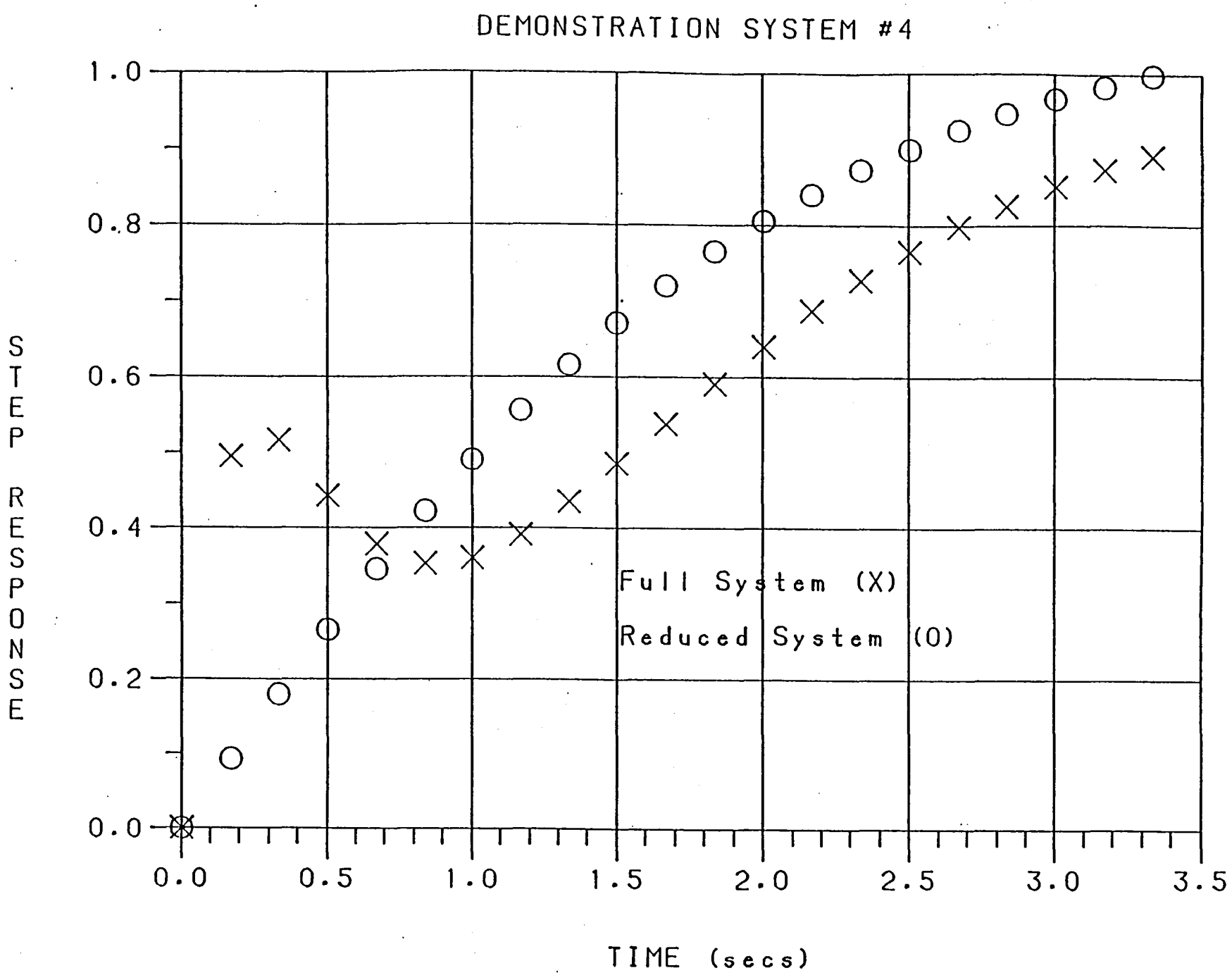


FIGURE 14 SUBSYSTEM ELIMINATION

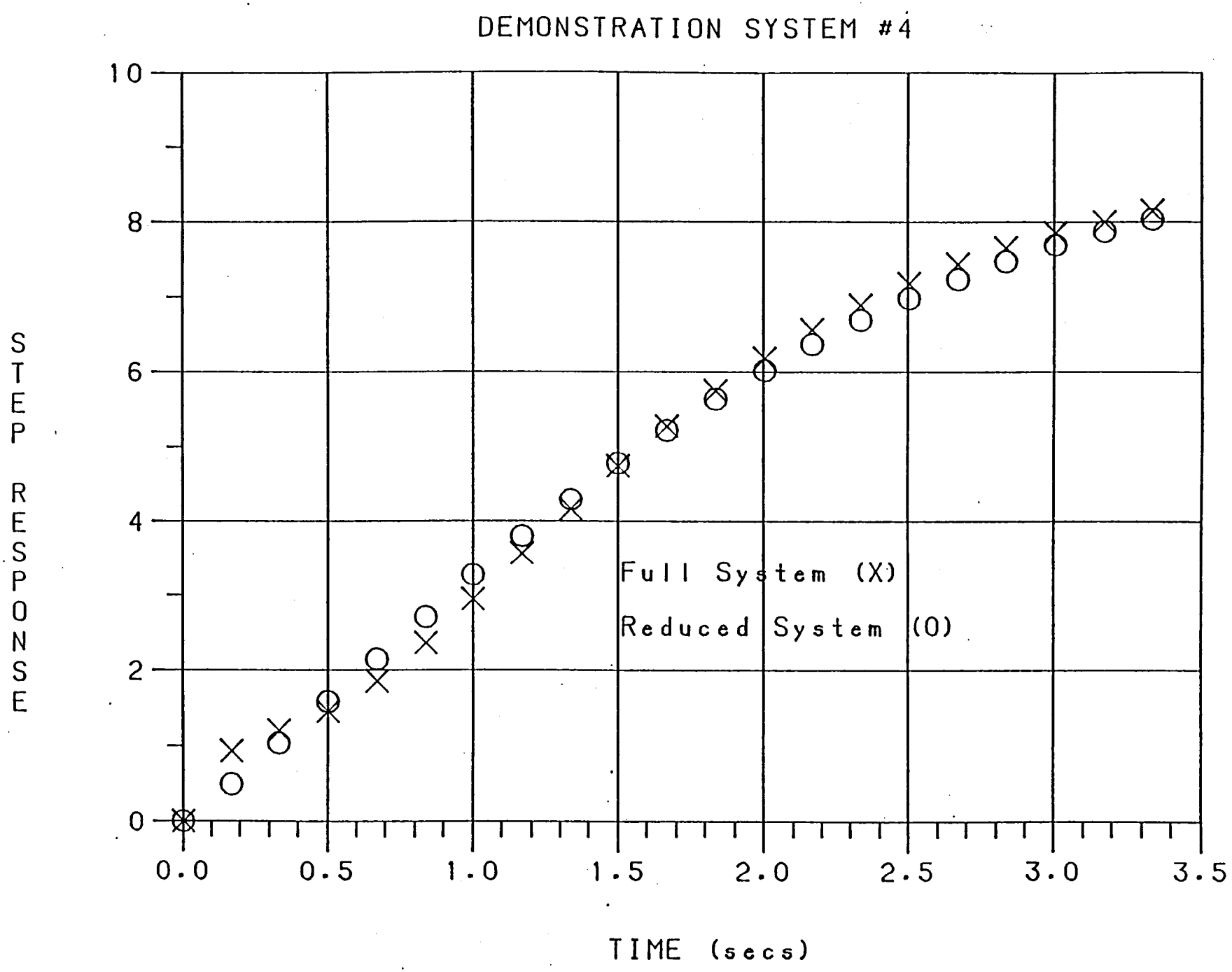


FIGURE 15 SUBSYSTEM ELIMINATION

MODRED USER'S GUIDE

Appendix B

Program Listings

```
//#####################################################################

//  FILENAME: BALANS.DAT
//  CREATED BY:K. R. DUNIPACE
//  DATE: 8/11/83
//  VERSION 4.2:  5/26/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PREPARES "BALANCED" SYSTEM MODEL OF CONTINUOUS SYSTEM FROM
//  STANDARD STATE VARIABLE MODEL;  XDOT = F X + G U; Y = H X + J U.
//  SYSTEM IS "BALANCED" IN SENSE OF B. C. MOORE REPORT "PRINCIPAL
//  COMPONENT ANALYSIS IN LINEAR SYSTEMS: ...",  IEEE AC-26, 1-81,
//  PP 21FF.  THE ALGORITHMS BELOW ARE IMPLEMENTATIONS OF FORMULAS
//  PRESENTED IN THE REPORT.

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  F - F MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           3)  G - G MATRIX  "   "      "     "     "
//           4)  H - H MATRIX  "   "      "     "     "
//           5)  J - J MATRIX  "   "      "     "     "
//           6)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           7)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE
//           1)  NPTSG - NUMBER OF SAMPLE PTS. IN GRAMMIAN
//           2)  TI - TIME INTERVAL FOR APPROX. GRAMMIAN
//  DATA OUTPUT:
//       A)  TO STACK
//           1)  FB - F MATRIX OF BALANCED SYSTEM
//           2)  GB - G MATRIX OF    "       "
//           3)  HB - H MATRIX OF    "       "
//           4)  IGN - EIGENVALUES OF ORIG. SYST.
//           5)  LAM - SINGULAR VALUES OF BAL. SYST.
//           6)  NS - ORDER OF ORIG. SYST.
//           7)  TI - TIME INTERVAL FOR SIMULATION.
//       B)  TO STORAGE FILES: (NONE)
//       C)  TO CONSOLE:
//           1)  FB - F MATRIX OF BALANCED SYSTEM
//           2)  GB - G MATRIX OF    "       "
//           3)  HB - H MATRIX OF    "       "
//           4)  IGN - EIGENVALUES OF ORIG. SYST.
//           5)  LAM - SINGULAR VALUES OF BAL. SYST.

//*********************************************************************

//     DEFINITION OF MACROS

//-------------------------------------------------------------

//   BEGIN MACRO DEFINITION FOR STABLE SYSTEM HERE.

//-------------------------------------------------------------

//   BEGIN "STAB" HERE.
//   THIS ALGORITHM WAS COPIED FROM "MATRIXX USER'S GUIDE"
//   (1982 ED. PP. 4-38 TO 4-39)
```

```
STAB='WC=LYAP(F,G*G'');..
WO=LYAP(F'',H''*H);..
LC=CHOL(WC);..
[V,D]=EIG(LC*WO*LC'');..
P=LC''*V*DIAG(DIAG(D)**(-.25));..
MSG=('' THE BALANCED MODEL IS:'');..
SKILMAC;..
FB=P\F*P,..
GB=P\G,..
HB=H*P,..
WAITMAC;..
MSG=('' THE COMPONENT MAGNITUDES ARE:'');..
SKILMAC;..
WBC=LYAP(FB,GB*GB'');..
LAM=SQRT(DIAG(WBC))';
//  DISPLAY THE COMPONENT MAGNITUDES FOR USE IN DETERMINING WHICH
//  SUBSYSTEMS SHOULD BE ELIMINATED IN THE REDUCTION PROCESS.
CLEAR WBC P V WO LC WC D;

//  END MACRO DEFINITION OF ALGORITHM FOR STABLE SYSTEM

//-------------------------------------------------------------
//-------------------------------------------------------------

//  BEGIN MACRO DEFINITIONS FOR UNSTABLE SYSTEM.

//-------------------------------------------------------------

//  THESE ALGORITHMS ARE AN IMPLEMENTATION OF EQUATIONS FROM
//  B. C. MOORE REPORT "PRINCIPAL COMPONENT ANALYSIS IN
//  LINEAR SYSTEMS: ...", IEEE AC-26, 1-81, PP21FF..

//-------------------------------------------------------------

//  BEGIN "GRAMC" HERE.
//  GRAMC PRODUCES THE RRSPONSES OF EACH STATE VARIABLE RESULTING
//  FROM AN IMPULSE AT EACH INPUT.  IT THEN REARRANGES SAMPLED
//  VALUES FROM THESE RESPONSES INTO A MATRIX DNT WHICH HAS
//  SINGULAR VALUES AND VECTORS WHICH ARE AYSMPTOTIC TO THE
//  SINGULAR VALUES AND VECTORS OF THE CONTROLLABILITY GRAMMIAN.

GRAMC='NSMI=NS*MI;..
DNT=QC(:,1:MI:NSMI);..
FOR I=2:MI,T1=QC(:,I:MI:NSMI);DNT=DNT;T1;END;';

//-------------------------------------------------------------

//  BEGIN "GRAMO" HERE.
//  GRAMO PRODUCES THE RESPONSE OF EACH OUTPUT VARIABLE TO A UNIT
//  INITIAL CONDITION OF EACH STATE VARIABLE.  IT THEN REARRANGES
//  SAMPLED VALUES FROM THESE RESPONSES INTO A MATRIX DNOT WHICH
//  HAS SINGULAR VALUES AND VECTORS WHICH ARE ASYMPTOTIC TO THE
//  SINGULAR VALUES AND VECTORS OF THE OBSERVABILITY GRAMMIAN.

GRAMO='KHNS=KH*NS;..
DNOT=QO(:,1:NS);..
FOR I=2:KH,T3=QO(:,1+NS*(I-1):NS*I);DNOT=DNOT;T3;END;';

//-------------------------------------------------------------
```

```
//  BEGIN "UNSTAB" HERE.
//  UNSTAB IS THE ALGORITHM FOR BALANCING AN UNSTABLE SYSTEM.

UNSTAB='SC=F,G;EYE(NS),0*ONES(NS,MI);..
MSG=('' THIS SYSTEM HAS UNSTABLE EIGENVALUES, THUS THE'');..
SKILMAC;..
MSG=('' GRAMMIAN IS ONLY A FINITE-TIME APPROXIMATION.'');..
SKILMAC;..
MSG=('' ENTER TIME INTERVAL AND NUMBER OF POINTS FOR'');..
SKILMAC;..
MSG=('' APPROXIMATING THE GRAMMIAN.'');..
SKILMAC;..
MSG=('' I. E. 0.2 AND 20'');..
SKILMAC;..
INQUIRE TI ''ENTER TIME INTERVAL TI'';..
INQUIRE NPTSG  ''ENTER NUMBER OF POINTS NPTSG'';..
COMPMAC;..
TX,QC=TIMR(SC,NS,TI,NPTSG);..
GRAMC;..
DUM,SIGC,VC=SVD(DNT,0);..
CLEAR GRAMC DUM QC SC DNT;..
DT=SQRT(TI/NPTSG);..
SIGC=DT*SIGC;..
P1=VC*SIGC;..
FSQ=P1\F*P1;..
GSQ=P1\G;..
HSQ=H*P1;..
CLEAR P1;..
KH,K=SIZE(HSQ);..
SO=FSQ,EYE(NS);HSQ,0*ONES(KH,NS);..
TX,QO=TIMR(SO,NS,TI,NPTSG);..
GRAMO;..
DUM,SIGO,VO=SVD(DNOT,0);..
CLEAR U QO SO GRAMO DUM DNOT;..
SIGO=DT*SIGO;..
P2=VO/SQRT(SIGO);..
MSG=('' THE BALANCED MODEL IS:'');..
SKILMAC;..
FB=P2\FSQ*P2,..
GB=P2\GSQ,..
HB=HSQ*P2,..
CLEAR FSQ GSQ HSQ;..
WAITMAC;..
MSG=('' THE COMPONENT MAGNITUDES ARE:'');..
SKILMAC;..
SHORT E;..
LAM=SVD(SIGO*VO''*VC*SIGC),..
SHORT;..
WC2U=P2\EYE(NS)/(P2'');';
//  WC2U IS NOT ACTUALLY USED.  IT IS A TEST OF THE BALANCING ALGORITHM.
//  WHEN THE BALANCING ALGORITHM IS WORKING PROPERLY WC2U SHOULD BE THE
//  IDENTITY MATRIX.  IT WAS USED DURING PROGRAM TESTING AND IS LEFT IN
//  THE PROGRAM FOR ARCHIVAL PURPOSES.
CLEAR P2 SIGO SIGC VO VC;

//  END OF MACROS FOR UNSTABLE ALGORITHM

//-----------------------------------------------------------------
```

```
//     END OF MACRO DEFINITIONS

//=====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=====================================================================

//  COMMAND FILE "BALANS" BEGINS EXECUTION HERE.

//  LOAD SYSTEM MODEL FROM "SYSB".
LOAD('SYSB');
K,NS=SIZE(F);
K,MI=SIZE(G);
P,D=EIG(F);
MSG=(' THE EIGENVALUES OF THE SYSTEM ARE:');
SKILMAC;
IGN=DIAG(D)
CLEAR P D;
WAITMAC;
COMPMAC;
//  TEST EIGENVALUES TO DETERMINE WHETHER TO USE "STABLE" OR
//  "UNSTABLE" ALGORITHM FOR BALANCING.
KK=0;
FOR K=1:NS, IF REAL(IGN(K))=0,KK=1;END;
IF KK=0,STAB;END;
IF KK=1,UNSTAB;END;

//  RETURN CONTROL TO "MOORE"

RETURN;

//=====================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=====================================================================

//  END OF COMMAND FILE "BALANS.DAT"

//#####################################################################


$
```

```
//#####################################################################

//  FILENAME:  BALDISC.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE: 7/18/84
//  VERSION 4.2:  6/23/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PREPARES "BALANCED" SYSTEM MODEL FROM STANDARD DISCRETE STATE
//  VARIABLE MODEL;  X(K+1) = F X(K) + G U(K); Y = H X(K) + J U(K).
//  SYSTEM IS "BALANCED" IN SENSE OF B. C. MOORE REPORT (BELOW).
//  THESE ALGORITHMS ARE AN IMPLEMENTATION OF EQUATIONS FROM
//  B. C. MOORE REPORT "PRINCIPAL COMPONENT ANALYSIS IN
//  LINEAR SYSTEMS: ...", IEEE AC-26, 1-81, PP21FF. EXTRAPOLATED
//  TO THE DISCRETE-TIME CASE.

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  F - F MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           3)  G - G MATRIX  "   "       "     "     "
//           4)  H - H MATRIX  "   "       "     "     "
//           5)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           6)  J - J MATRIX  "   "       "     "     "
//           7)  TS - MODEL SAMPLING INTERVAL   "     "
//           8)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE
//           1)  NPTSG - NUMBER OF SAMPLE PTS. IN GRAMMIAN
//  DATA OUTPUT:
//       A)  TO STACK
//           1)  FB - F MATRIX OF BALANCED SYSTEM
//           2)  GB - G MATRIX OF    "       "
//           3)  HB - H MATRIX OF    "       "
//           4)  IGN - EIGENVALUES OF ORIG. SYST.
//           5)  LAM - SINGULAR VALUES OF BAL. SYST.
//           6)  NS - ORDER OF ORIG. SYST.
//           7)  TB - TIME VECTOR FOR PLOTTING TIME RESP.
//           8)  TS - SAMPLING INTERVAL OF DISC.-TIME MODEL
//       B)  TO STORAGE FILES: (NONE)
//       C)  TO CONSOLE:
//           1)  IGN - EIGENVALUES OF ORIG. SYST.
//           2)  LAM - SINGULAR VALUES OF BAL. SYST.

//*********************************************************************

//     DEFINITION OF MACROS

//-------------------------------------------------------------------

//  BEGIN MACRO DEFINITIONS FOR DISCRETE SYSTEM HERE

//-------------------------------------------------------------------

//  BEGIN "GRAMC" HERE.
//  GRAMC REARRANGES THE IMPULSE RESPONSES OF THE STATE
//  VARIABLES INTO A MATRIX "DNT" WHICH HAS THE SAME SINGULAR
//  VALUES AND VECTORS AS THE CONTROLLABILITY GRAMMIAN.
```

```
GRAMC='DNT=QC(:,1:NS);..
     FOR I=2:MI,DNT=[DNT;[QC(:,1+NS*(I-1):NS*I)]];END;';

//-------------------------------------------------------------

//  BEGIN "GRAMO" HERE.
//  GRAMO REARRANGES THE INITIAL CONDITION RESPONSES OF THE
//  OUTPUTS INTO A MATRIX "DNOT" WHICH HAS THE SAME SINGULAR
//  VALUES AND VECTORS AS THE OBSERVABILITY GRAMMIAN.

GRAMO='KHNS=KH*NS;DNOT=QO(:,1:KH:KHNS);..
     FOR I=2:KH,DNOT=[DNOT;[QO(:,I:KH:KHNS)]];END;';

//-------------------------------------------------------------

//  BEGIN "QCMAC" HERE.
//  QCMAC PRODUCES THE RESPONSE OF EACH STATE VARIABLE RESULTING
//  FROM AN IMPULSE AT EACH INPUT.  THE RESPONSES ARE USED BY
//  "GRAMC" ABOVE.

QCMAC='FOR I=1:MI,..
     U=0*ONES(NPTSG,MI);..
     U(1,I)=1/TS;..
     QC=[QC,[FILP(SC,U)]];END;';

//-------------------------------------------------------------

//  BEGIN "QOMAC" HERE.
//  QOMAC PRODUCES THE RESPONSE OF EACH OUTPUT VARIABLE
//  RESULTING FROM A UNIT INITIAL CONDITION OF EACH STATE
//  VARIABLE.  THE RESPONSES ARE USED IN "GRAMO" ABOVE.

QOMAC='FOR I=1:NS;..
     U=0*ONES(NPTSG,NS);..
     U(1,I)=1/TS;..
     QO=[QO,[FILP(SO,U)]];END;';

//-------------------------------------------------------------

//     END OF MACRO DEFINITIONS

//=============================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=============================================================

//  COMMAND FILE "BALDISC" BEGINS EXECUTION HERE.

//  LOAD SYSTEM MODEL FROM FILE "SYSB"
LOAD('SYSB');
//  DETERMINE SIZE AND EIGENVALUES OF SYSTEM
K,NS=SIZE(F);
K,MI=SIZE(G);
DUM,D=EIG(F);
CLEAR DUM;
MSG=(' THE EIGENVALUES OF THE SYSTEM ARE:');
SKILMAC;
IGN=DIAG(D)
CLEAR D;
WAITMAC;
```

```
// BEGIN BALANCING ALGORITHM

MSG=(..
'  THIS SYSTEM IS DISCRETE, THUS THE GRAMMIAN IS ONLY A    ';
'  FINITE-TIME APPROXIMATION.  DEFINE SAMPLING POINTS FOR ';
'  APPROXIMATING THE GRAMMIAN.  E. G. 20                   ');
SKILMAC;
MSG=('THE SAMPLING INTERVAL IS:');
SKILMAC;
TS
INQUIRE NPTSG 'ENTER NUMBER OF POINTS NPTSG';
COMPMAC;
//  MAKE UP TIME VECTOR FOR USE IN PLOTTING.
TB=TS*[0:NPTSG-1]';
//  MAKE UP "SYSTEM" TO DISPLAY STATE VARIABLES AS OUTPUT
SC=F,G;EYE(NS),0*ONES(NS,MI);
//  CALCULATE MATRIX FOR CONTOLLABILITY TRANSFORMATION OF
//  BALANCING ALGORITHM.
QC=ONES(NPTSG,NS);
QCMAC;
CLEAR QCMAC;
QC=[QC(:,NS+1:NS*(MI+1))];
CLEAR SC;
GRAMC;
CLEAR GRAMC;
DUM,SIGC,VC=SVD(DNT,0);
CLEAR DUM QC DNT;
P1=VC*SIGC;
//  PERFORM CONTROLLABILITY TRANSFORMATION.
FSQ=P1\F*P1;
GSQ=P1\G;
HSQ=H*P1;
CLEAR P1;
//  CALCULATE MATRIX FOR OBSERVABILITY TRANSFORMATION OF
//  BALANCING ALGORITHM.
KH,K=SIZE(HSQ);
SO=FSQ,EYE(NS);HSQ,0*ONES(KH,NS);
U=[ONES(1,NS);0*ONES(NPTSG-1,NS)];
QO=ONES(NPTSG,KH);
QOMAC;
CLEAR QOMAC;
QO=[QO(:,KH+1:KH*(NS+1))];
GRAMO;
CLEAR U QO SO GRAMO;
DUM,SIGO,VO=SVD(DNOT,0);
CLEAR DUM DNOT;
P2=VO/SQRT(SIGO);
//  PERFORM OBSERVABILITY TRANSFORMATION - OBTAIN BALANCED MODEL.
MSG=('  THE BALANCED MODEL IS:');
SKILMAC;
FB=P2\FSQ*P2
GB=P2\GSQ
HB=HSQ*P2
CLEAR FSQ GSQ HSQ;
WAITMAC;
MSG=('  THE COMPONENT MAGNITUDES ARE:');
SKILMAC;
SHORT E;
LAM=SVD(SIGO*VO'*VC*SIGC)
```

```
CLEAR VC VO;
SHORT;
CLEAR P2 K MI SIGC KH KHNS NPTSG SIGO;

//  RETURN CONTROL TO "DISCRT"

RETURN;


//=====================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=====================================================================

//  END OF COMMAND FILE "BALDISC.DAT"

//#####################################################################


$
```

```
//#########################################################################

//  FILENAME:  BALDISC.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE: 7/18/84
//  VERSION 4.2:  6/23/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PREPARES "BALANCED" SYSTEM MODEL FROM STANDARD DISCRETE STATE
//  VARIABLE MODEL;  X(K+1) = F X(K) + G U(K); Y = H X(K) + J U(K).
//  SYSTEM IS "BALANCED" IN SENSE OF B. C. MOORE REPORT (BELOW).
//  THESE ALGORITHMS ARE AN IMPLEMENTATION OF EQUATIONS FROM
//  B. C. MOORE REPORT "PRINCIPAL COMPONENT ANALYSIS IN
//  LINEAR SYSTEMS: ...", IEEE AC-26, 1-81, PP21FF. EXTRAPOLATED
//  TO THE DISCRETE-TIME CASE.

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  F - F MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           3)  G - G MATRIX  "   "       "     "     "
//           4)  H - H MATRIX  "   "       "     "     "
//           5)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           6)  J - J MATRIX  "   "       "     "     "
//           7)  TS - MODEL SAMPLING INTERVAL   "     "
//           8)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE
//           1)  NPTSG - NUMBER OF SAMPLE PTS. IN GRAMMIAN
//  DATA OUTPUT:
//       A)  TO STACK
//           1)  FB - F MATRIX OF BALANCED SYSTEM
//           2)  GB - G MATRIX OF    "       "
//           3)  HB - H MATRIX OF    "       "
//           4)  IGN - EIGENVALUES OF ORIG. SYST.
//           5)  LAM - SINGULAR VALUES OF BAL. SYST.
//           6)  NS - ORDER OF ORIG. SYST.
//           7)  TB - TIME VECTOR FOR PLOTTING TIME RESP.
//           8)  TS - SAMPLING INTERVAL OF DISC.-TIME MODEL
//       B)  TO STORAGE FILES: (NONE)
//       C)  TO CONSOLE:
//           1)  IGN - EIGENVALUES OF ORIG. SYST.
//           2)  LAM - SINGULAR VALUES OF BAL. SYST.

//*************************************************************************

//     DEFINITION OF MACROS

//-------------------------------------------------------------------------

//  BEGIN MACRO DEFINITIONS FOR DISCRETE SYSTEM HERE

//-------------------------------------------------------------

//  BEGIN "GRAMC" HERE.
//  GRAMC REARRANGES THE IMPULSE RESPONSES OF THE STATE
//  VARIABLES INTO A MATRIX "DNT" WHICH HAS THE SAME SINGULAR
//  VALUES AND VECTORS AS THE CONTROLLABILITY GRAMMIAN.
```

```
GRAMC='DNT=QC(:,1:NS);..
     FOR I=2:MI,DNT=[DNT;[QC(:,1+NS*(I-1):NS*I)]];END;';

//-----------------------------------------------------------------

//  BEGIN "GRAMO" HERE.
//  GRAMO REARRANGES THE INITIAL CONDITION RESPONSES OF THE
//  OUTPUTS INTO A MATRIX "DNOT" WHICH HAS THE SAME SINGULAR
//  VALUES AND VECTORS AS THE OBSERVABILITY GRAMMIAN.

GRAMO='KHNS=KH*NS;DNOT=QO(:,1:KH:KHNS);..
     FOR I=2:KH,DNOT=[DNOT;[QO(:,I:KH:KHNS)]];END;';

//-----------------------------------------------------------------

//  BEGIN "QCMAC" HERE.
//  QCMAC PRODUCES THE RESPONSE OF EACH STATE VARIABLE RESULTING
//  FROM AN IMPULSE AT EACH INPUT.  THE RESPONSES ARE USED BY
//  "GRAMC" ABOVE.

QCMAC='FOR I=1:MI,..
     U=0*ONES(NPTSG,MI);..
     U(1,I)=1/TS;..
     QC=[QC,[FILP(SC,U)]];END;';

//-----------------------------------------------------------------

//  BEGIN "QOMAC" HERE.
//  QOMAC PRODUCES THE RESPONSE OF EACH OUTPUT VARIABLE
//  RESULTING FROM A UNIT INITIAL CONDITION OF EACH STATE
//  VARIABLE.  THE RESPONSES ARE USED IN "GRAMO" ABOVE.

QOMAC='FOR I=1:NS;..
     U=0*ONES(NPTSG,NS);..
     U(1,I)=1/TS;..
     QO=[QO,[FILP(SO,U)]];END;';

//-----------------------------------------------------------------

//     END OF MACRO DEFINITIONS

//=================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=================================================================

//  COMMAND FILE "BALDISC" BEGINS EXECUTION HERE.

//  LOAD SYSTEM MODEL FROM FILE "SYSB"
LOAD('SYSB');
//  DETERMINE SIZE AND EIGENVALUES OF SYSTEM
K,NS=SIZE(F);
K,MI=SIZE(G);
DUM,D=EIG(F);
CLEAR DUM;
MSG=(' THE EIGENVALUES OF THE SYSTEM ARE:');
SKILMAC;
IGN=DIAG(D)
CLEAR D;
WAITMAC;
```

```
// BEGIN BALANCING ALGORITHM

MSG=(..
'  THIS SYSTEM IS DISCRETE, THUS THE GRAMMIAN IS ONLY A    ';
'  FINITE-TIME APPROXIMATION.  DEFINE SAMPLING POINTS FOR ';
'  APPROXIMATING THE GRAMMIAN.  E. G. 20                   ');
SKILMAC;
MSG=('THE SAMPLING INTERVAL IS:');
SKILMAC;
TS
INQUIRE NPTSG 'ENTER NUMBER OF POINTS NPTSG';
COMPMAC;
//  MAKE UP TIME VECTOR FOR USE IN PLOTTING.
TB=TS*[0:NPTSG-1]';
//  MAKE UP "SYSTEM" TO DISPLAY STATE VARIABLES AS OUTPUT
SC=F,G;EYE(NS),0*ONES(NS,MI);
//  CALCULATE MATRIX FOR CONTOLLABILITY TRANSFORMATION OF
//  BALANCING ALGORITHM.
QC=ONES(NPTSG,NS);
QCMAC;
CLEAR QCMAC;
QC=[QC(:,NS+1:NS*(MI+1))];
CLEAR SC;
GRAMC;
CLEAR GRAMC;
DUM,SIGC,VC=SVD(DNT,0);
CLEAR DUM QC DNT;
P1=VC*SIGC;
//  PERFORM CONTROLLABILITY TRANSFORMATION.
FSQ=P1\F*P1;
GSQ=P1\G;
HSQ=H*P1;
CLEAR P1;
//  CALCULATE MATRIX FOR OBSERVABILITY TRANSFORMATION OF
//  BALANCING ALGORITHM.
KH,K=SIZE(HSQ);
SO=FSQ,EYE(NS);HSQ,0*ONES(KH,NS);
U=[ONES(1,NS);0*ONES(NPTSG-1,NS)];
QO=ONES(NPTSG,KH);
QOMAC;
CLEAR QOMAC;
QO=[QO(:,KH+1:KH*(NS+1))];
GRAMO;
CLEAR U QO SO GRAMO;
DUM,SIGO,VO=SVD(DNOT,0);
CLEAR DUM DNOT;
P2=VO/SQRT(SIGO);
//  PERFORM OBSERVABILITY TRANSFORMATION - OBTAIN BALANCED MODEL.
MSG=('  THE BALANCED MODEL IS:');
SKILMAC;
FB=P2\FSQ*P2
GB=P2\GSQ
HB=HSQ*P2
CLEAR FSQ GSQ HSQ;
WAITMAC;
MSG=('  THE COMPONENT MAGNITUDES ARE:');
SKILMAC;
SHORT E;
LAM=SVD(SIGO*VO'*VC*SIGC)
```

```
CLEAR VC VO;
SHORT;
CLEAR P2 K MI SIGC KH KHNS NPTSG SIGO;

//  RETURN CONTROL TO "DISCRT"

RETURN;


//=====================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=====================================================================

//  END OF COMMAND FILE "BALDISC.DAT"

//#####################################################################


$
```

```
//#####################################################################

//
//  FILENAME:  DIFPN.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE:  SUMMER '83
//  VERSION 4.2:  8/5/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PROGRAM PERFORMS MODEL REDUCTION OF CONTINUOUS SINGLE-INPUT
//  SINGLE-OUTPUT SYSTEM REPRESENTED BY A TRANSFER FUNCTION.
//  METHOD IS "DIFFERENTIATION OF POLYNOMIALS" DESCRIBED IN
//  "CONTRIBUTIONS TO THE MODEL REDUCTION PROBLEM" BY GUTMAN,
//  MANNERFELT, AND MOLANDER IEEE AC-27, NO. 2, 4-82, PP454-455.

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES:
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           3)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE:
//           1)  AI - DENOMINATOR COEFFICIENTS OF TRANS. FN.
//           2)  BI - NUMERATOR COEFFICIENTS OF TRANS. FN.
//           3)  NPTS - NUMBER OF POINTS IN SIMULATION FOR PLOTTING
//           4)  NSR - ORDER OF REDUCED MODEL DESIRED FROM REDUCTION
//           5)  T - TIME DURATION FOR SIMULATION
//  DATA OUTPUT:
//       A)  TO STACK:  (ALL VARIABLES - NONE CLEARED)
//       B)  TO STORAGE FILES: (NONE)
//       C)  TO CONSOLE:
//           1)  DR - DENOMINATOR COEFFS. OF REDUCED TRANS. FN.
//           2)  ERR - RMS ERROR OF REDUCED STEP RESPONSE
//           3)  NR - NUMERATOR COEFFS. OF REDUCED TRANS. FN.
//           4)  STR - STEP RESPONSE OF BOTH SYSTEMS AND ERROR

//=====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=====================================================================

//  COMMAND FILE "DIFPN" BEGINS EXECUTION HERE.

//
//  ENTER THE COEFFICIENTS OF TRANSFER FUNCTION.
//
MSG=('ENTER COEFFICIENTS BI, OF NUMERATOR POLYNOMIAL.');
SKILMAC;
MSG=('E.G. 0.48,4.8,12');
SKILMAC;
INQUIRE BI 'ENTER BI';
MSG=('ENTER COEFFICIENTS OF DENOMINATOR POLYNOMIAL.');
SKILMAC;
MSG=('N.B. COEFF. OF ""S**N"" (NOT ENTERED) MUST BE 1.');
SKILMAC;
MSG=('E.G. 9,20,12');
SKILMAC;
INQUIRE AI 'ENTER AI';
COMPMAC;
```

```
//  MAKE UP STATE-VARIABLE MODEL, IN OBSERVER-CANONICAL FORM,
//  FOR SIMULATION OF STEP RESPONSE.  ESSENTIALLY SAME AS
//  ALGORITHM PRESENTED IN "MATRIXX USER'S GUIDE" VERSION 3.0
//  PAGE 4-6.
I,NS=SIZE(AI);
I,MI=SIZE(BI);
C=0;
ON1=ONES(NS-1,1);
F=DIAG(ON1,1);
F(:,1)=-AI(1:NS)';
G=0*ONES(1,NS-MI),BI';
H=[1,0*ON1'];
J=C;
SE=[F,G,0*G;0*G',0,1;H,J,0];
//  INPUT THE ORDER DESIRED IN REDUCED SYSTEM
MSG=(' ENTER NSR, THE ORDER FOR THE REDUCED SYSTEM.  E.G. 2');
SKILMAC;
INQUIRE NSR 'ENTER NSR';
COMPMAC;
//  CALCULATE DERIVATIVES OF  POLYNOMIALS.  ALGORITHM IS
//  DIRECT IMPLEMENTATION OF STANDARD FORMULA FROM MATH TABLES.
//  ORDER OF DENOMINATOR
NSD = NS;
//  ORDER OF NUMERATOR
NSN = MI-1;
//  DENOMINATOR COEFFICIENTS (EXCEPT FIRST WHICH IS 1)
PDR=AI;
//  NUMERATOR COEFFICIENTS
PNR=BI;
//  PDR IS REDUCED DENOMINATOR, PNR IS REDUCED NUMERATOR.
FOR JK=1:NS-NSR;...
   FOR KD=1:NS-JK;PDR(KD)=PDR(KD)*(1-(NSD-KD)/NSD);END;...
   K1=PDR(1);...
   NSD=NSD-1;...
   FOR JJ=1:NSD;PDR(JJ)=PDR(JJ+1)/K1;END;...
   FOR KN=1:MI-JK-1;PNR(KN)=PNR(KN+1)*(1-(NSN-KN)/NSN)/K1;END;...
   PNR(NSN)=PNR(NSN+1)/K1;...
NSN = NSN-1;END;
//  MAKE UP STATE-VARIABLE MODEL OF REDUCED TRANSFER FUNCTION FOR
//  SIMULATION OF STEP RESPONSE.
IF NSR1;ON2=ONES(NSR-1,1);FR=DIAG(ON2,1);FR(:,1)=-PDR(1:NSR)';...
   HR=[1,0*ON2'];END;
IF NSR=1;FR=-PDR(1);HR=1;END;
GR=PNR(1:NSR)';
JR=J;
SER=[FR,GR,0*GR;0*GR',0,1;HR,0,J];
//  INPUT TIME DURATION AND NUMBER OF POINTS DISPLAYED IN SIMULATION
MSG=(' ENTER T, TIME DURATION FOR SIMULATION.  E.G. 3.5');
SKILMAC;
INQUIRE T ;'ENTER T';
MSG=(' ENTER NPTS, NUMBER OF POINTS FOR SIMULATION.  E.G. 20');
SKILMAC;
INQUIRE NPTS 'ENTER NPTS';
COMPMAC;
//  MAKE UP TIME RESPONSES AND ERROR FOR PLOTTING.
[TD,STRF]=TIMR(SE,NS+1,[T,NPTS]);
[TR,STRR]=TIMR(SER,NSR+1,[T,NPTS]);
STRE=STRF-STRR;
STR=[STRF,STRR,STRE];
ERR=0;
```

```
ERR=ERR+SUM(STRE.*STRE/SUM(STRF.*STRF));
CLEAR STRE STRF STRR;
MSG=('  --  WHEN FINISHED WITH PLOT, NOT NOW, CR TO CONT.  --');
SKILMAC;
PLOT(TR,STR,'YLAB/STEP RESPONSE/ XLAB/TIME [SEC]/ TITL/MODEL ...
REDUCTION BY DIFFERENTIATION OF POLYNOMIALS/ TIC2 FULL ');
PAUSE;
PLOT('CLEAR')
//  CALCULATE AND DISPLAY RMS ERROR
MSG=(' THE RMS ERROR FOR THIS REDUCTION IS:');
SKILMAC;
ERR=SQRT(ERR/NPTS)
//  DISPLAY COEFFICIENTS OF REDUCED TRANSFER FUNCTION.
MSG=(' IN THE REDUCED TRANSFER FUNCTION,');
SKILMAC;
MSG=(' THE COEFFICIENTS OF THE NUMERATOR POLYNOMIAL ARE:');
SKILMAC;
NR=GR
MSG=(' THE COEFFICIENTS OF THE DENOMINATOR POLYNOMIAL ARE:');
SKILMAC;
DR=-FR(:,1)'
WAITMAC;

//  RETURN CONTROL TO "MENU"

EXEC('MENU')

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "DIFPN.DAT"

//#################################################################

$
```

```
//#####################################################################

//  FILENAME:  DISCRT.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE: 7/18/84
//  VERSION 4.2: 5/27/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PERFORMS MODEL REDUCTION OF DISCRETE SYSTEMS BY "SUBSYSTEM
//  ELIMINATION" AS PRESENTED BY B. C. MOORE IN "PRINCIPAL COMPONENT
//  ANALYSIS IN LINEAR SYSTEMS, ..." IEEE AC-26, 1-81.
//  MODEL IS STANDARD STATE VARIABLE MODEL:
//           X(K+1) = F X(K) + G U(K);
//           Y(K) = H X(K) + J U(K).

//  DATA INPUT:
//      A)  FROM OTHER PROGRAMS AND FILES
//          1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//          1)  F - F MATRIX OF ORIG. SYST. FROM "BALDISC"
//          2)  FB - BALANCED F MATRIX          "      "
//          3)  G - G MATRIX OF ORIG. SYST.     "      "
//          4)  GB - BALANCED G MATRIX          "      "
//          5)  H - H MATRIX OF ORIG. SYST.     "      "
//          6)  HB - BALANCED H MATRIX          "      "
//          7)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//          8)  J - J MATRIX OF BOTH SYSTS.     "      "
//      B)  FROM CONSOLE
//          1)  RCOL - COMPONENT SELECTION VECTOR
//          2)  TS - SAMPLING INTERVAL FROM MODEL
//          3)  NPTS - NO. OF POINTS FOR SIMULATION
//  DATA OUTPUT:
//      A)  TO STACK:
//          1)  FBR - F MATRIX OF BALANCED REDUCED SYSTEM
//          2)  GBR - G MATRIX OF BALANCED REDUCED SYSTEM
//          3)  HBR - H MATRIX OF BALANCED REDUCED SYSTEM
//          4)  NPTS - NO. OF POINTS IN SIMULATION
//          5)  NSR - ORDER OF REDUCED MODEL.
//          6)  RCOL - COMPONENT SELECTION VECTOR
//          7)  STR - STEP RESPONSE MATRIX FOR PLOTTING
//          8)  TB - TIME VECTOR FOR PLOTTING
//          9)  TS - SAMPLING INTERVAL FROM MODEL
//      B)  TO STORAGE FILES:  (NONE)
//      C)  TO CONSOLE:
//          1)  PLOTS OF STEP RESPONSES

//====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//====================================================================

//  COMMAND FILE "DISCRT" BEGINS EXECUTION HERE.

//  BRANCH TO "BALDISC" TO DETERMINE BALANCED MODEL

EXEC('BALDISC');

//  RETURN FROM "BALDISC"
```

```
MSG=('SELECT COMPONENTS TO BE RETAINED.                    ';
      'ENTER, IN ORDER, ""0"" OR ""1"" FOR EACH COMPONENT.';
      '""1"" RETAINS COMPONENT; ""0"" DELETES.              ';
      'E.G.:  1,1,0                                      ');
SKILMAC;
INQUIRE RCOL 'ENTER RCOL';
MSG=('ENTER NO. OF POINTS FOR SIMULATION.  E.G.: 20');
SKILMAC;
INQUIRE NPTS 'ENTER NPTS';
COMPMAC;

//  FORM VECTOR OF TIME POINTS FOR TIME AXIS OF DISCRETE-TIME
//  RESPONSE (WHICH WOULD OTHERWISE BE BASED ON POINT NUMBER).
TB=TS*[0:NPTS-1]';

//  SCALE FACTOR FOR IMPULSE FUNCTION: TS*DTI = 1.0
DTI=1/TS;

//-----------------------------------------------------------------

//  FORM "SYSTEM" MATRIX FOR SIMULATION OF STEP RESPONSE OF
//  ORIGINAL SYSTEM.  REF:  MATRIXX USER'S GUIDE PG 4-8, 1/84 ED.

I,GCOL=SIZE(G);
HROW,I=SIZE(H);
SE=[F,G;H,J];
SE=REAL(SE);

//  FORM MATRIX OF STEP RESPONSES, ONE FOR EACH COMBINATION OF
//  INPUTS AND OUTPUTS.  START WITH DUMMY MATRIX, CONCATENATE,
//  THEN TRUNCATE.

//  DUMMY MATRIX
TRB=ONES(NPTS,HROW);

//  FORM STEP FUNCTIONS, ITERATIVELY, FOR EACH INPUT
//  CONCATENATE STEP RESPONSE FUNCTIONS FOR ALL OUTPUTS FROM
//  PRESENT INPUT TO EXISTING RESPONSE MATRIX.  START WITH DUMMY.
DTIV=ONES(NPTS,1);
FOR I=1:GCOL;..
U=0*ONES(NPTS,GCOL);..
U(:,I)=DTIV;..
TRB=[TRB,[FILP(SE,U)]];END;

//  REMOVE DUMMY MATRIX
TRB=[TRB(:,HROW+1:HROW*(GCOL+1))];
CLEAR SE;

//-----------------------------------------------------------------

//  FORM "SYSTEM" MATRIX FOR SIMULATION OF STEP RESPONSE OF
//  REDUCED SYSTEM.  SIMILAR TO ORIG. SYST. ABOVE.

//  FORM REDUCED MATRICES ACCORDING TO "RCOL" ENTRIES.
NR=0;
FOR I=1:NS;..
IF RCOL(I)=1;NR=NR+1;GBR(NR,:)=GB(I,:);HBR(:,NR)=HB(:,I);....
NC=0;...
FOR IC=1:NS;IF RCOL(IC)=1;NC=NC+1;FBR(NR,NC)=FB(I,IC);END
CLEAR NS NR NC;
```

```
NSR=SUM(RCOL);

//  FORM SYSTEM MATRIX
SBR=[FBR,GBR;HBR,J];
SBR=REAL(SBR);
TRBR=ONES(NPTS,HROW);

//  FORM STEP FUNCTIONS, ITERATIVELY, FOR EACH INPUT
//  CONCATENATE STEP RESPONSE FUNCTIONS FOR ALL OUTPUTS FROM
//  PRESENT INPUT TO EXISTING RESPONSE MATRIX.  START WITH DUMMY.
FOR I=1:GCOL;..
U=0*ONES(NPTS,GCOL);..
U(:,I)=DTIV;..
TRBR=[TRBR,[FILP(SBR,U)]];END;

//  REMOVE DUMMY MATRIX
TRBR=[TRBR(:,HROW+1:HROW*(GCOL+1))];
CLEAR HROW;
CLEAR SBR U DTIV;

//-----------------------------------------------------------------

//  FORM ERROR MATRIX
STRE=TRB-TRBR;

//  FORM AND PLOT TIME RESPONSE MATRIX
STR=[TRB,TRBR,STRE];
CLEAR TRBR ;
MSG=('  --  WHEN FINISHED VIEWING PLOT, NOT NOW, CR TO CONT.  --');
SKILMAC;
RH,CH=SIZE(HBR);
CLEAR CH;
FOR IC=1:RH*GCOL,PLOT(TB,STR(:,IC:RH*GCOL:3*RH*GCOL), ...
'YLAB/STEP RESPONSE/  XLAB/TIME SEC/ SYMBOL...
 TITL/MODEL REDUCTION BY SUBSYSTEM ELIMINATION/TIC2 GRID=1 FULL');...
PAUSE;END;
PLOT('CLEAR')

//  CALCULATE THE RMS ERROR OF RESPONSES
MSG=(' THE RMS ERROR FOR THIS REDUCTION IS:');
SKILMAC;
ERR=0;
ERR=ERR+SUM(STRE.*STRE/SUM(TRB.*TRB));
ERR=SQRT(ERR/(NPTS*RH*GCOL))
CLEAR ERR STRE TRB;

//  BRANCH TO OUTPUT ROUTINE TO DISPLAY REDUCED MODEL

IF RH+GCOL=2;EXEC('SING');ELSE EXEC('MULTI');END;END;
CLEAR RH GCOL;

//  RETURN FROM OUTPUT ROUTINE

//  RETURN CONTROL TO "MENU"

EXEC('MENU');

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================
```

```
//  END OF COMMAND FILE "DISCRT.DAT"
//#####################################################################

$
```

```
//#####################################################################

//  FILENAME: INDEX.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE:  8/7/85
//  VERSION 4.2:
//

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PROGRAM PROVIDES AN INTERACTIVE INDEX OF SYSTEM MODELS
//  WHICH CAN BE VIEWED FROM WITHIN "MODRED".
//  LISTINGS HERE ARE VERY BRIEF.  COMPLETE MODEL DEFINITION IS
//  CONTAINED IN THE USER MANUAL.

//====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//====================================================================

//  COMMAND FILE "INDEX" BEGINS EXECUTION HERE.

DISP('  THE MODELS LISTED BELOW ARE AVAILABLE AS DATA FILES FOR');
DISP('  PRACTICE IN USING AND INTERPRETING VARIOUS MODEL REDUCTION');
DISP('  METHODS.  TRANSFER FUNCTION MODELS MUST BE RECORDED AND');
DISP('  ENTERED MANUALLY AT THE PROGRAM PROMPTS.  STATE-VARIABLE');
DISP('  MODELS MUST BE LOADED INTO ""SYSB"" (NORMALLY THROUGH THE');
DISP('  ""LOAD"" OPTION OF THE MAIN MENU).  COMPLETE MODEL DEFINITIONS');
DISP('  ARE CONTAINED IN APPENDIX C OF THE USER''S GUIDE.');

DISP('  A MODEL FILENAME IS CONSTRUCTED BY CONCATENATING ""SYS"" WITH');
DISP('  A SYSTEM NUMBER AND A VERSION SYMBOL (IF APPLICABLE).');
DISP('  E.G. SYS1, SYS1S, OR SYS1D');
WAITMAC
DISP([..
'                 SYSTEM INDEX                                    ';..
'  SYST NO.    VERSIONS          CHARACTERISTICS                  ';..
'  1             .SD         SISO,REAL ROOTS,TRIVIAL RESIDUE,*    ';..
'  2             .           SISO,REAL ROOTS,EQUAL RESIDUES       ';..
'  2A            .SD         SISO,REAL ROOTS,EQUAL RESIDUES,*     ';..
'  3A            .S          SISO,REAL ROOTS,TRIVIAL RESIDUE      ';..
'  3B            .SD         SISO,REAL ROOTS,TRIVIAL RESIDUE,*    ';..
'  4              S          MIMO,REAL ROOTS,TRIVIAL RESIDUE      ';..
'  4A             SD         MIMO,REAL ROOTS,TRIVIAL RESIDUE,*    ';..
'  5             .SD         SISO,REAL ROOTS,UNSTABLE,*           ';..
'  6              SD         MISO,REAL ROOTS,UNSTABLE,*           ';..
'  7              SD         MIMO,REAL ROOTS,UNSTABLE,*           ';..
'  9               D         SISO,COMPLEX ROOTS,UNSTABLE,*        ';..
'  F4             SD         MIMO,COMPLEX,UNSTABLE,5TH ORDER      ';..
'  F15            SD         MIMO,COMPLEX,UNSTABLE,10TH ORDER     ';..
'  X29             D         MIMO,COMPLEX,UNSTABLE,4TH,45TH ORDER';..
'               KEY TO VERSION SYMBOLS                            ';..
'   .  BASIC TRANSFER FUNCTION MODEL (. NOT IN FILENAME)          ';..
'  S   CONTINUOUS-TIME STATE-VARIABLE MODEL                       ';..
'  D   DISCRETE-TIME STATE-VARIABLE MODEL                         ';..
'  *   DOMINANT TIME CONSTANT 1 SEC.                              ']);
WAITMAC

//  RETURN CONTROL TO "MENU"
```

```
EXEC('MENU');

//=====================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=====================================================================

//  END OF COMMAND FILE "INDEX.DAT"

//#####################################################################

$
```

```
//#################################################################

//  FILENAME:  LOAD.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE: 7/24/84
//  VERSION 4.2:  8/5/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  THIS COMMAND FILE LOADS A DATA FILE INTO A FILE NAMED
//  'SYSB' FOR USE IN "MODRED".  DATA FILE SHOULD CONTAIN AN
//  APPROPRIATE STATE VARIABLE MODEL OF A SYSTEM IN STANDARD STATE-
//  VARIABLE FORM:  XDOT(T) = F X(T) + G U(T); Y(T) = H X(T) + J U(T)
//  FOR CONTINUOUS SYSTEMS OR: X(K+1) = F X(K) = G U(K);
//  Y(K) = H X(K) + J U(K) FOR DISCRETE SYSTEMS.  SYSTEM MAY BE STABLE
//  OR UNSTABLE.  A 59TH ORDER SYSTEM WAS RUN SUCCESSFULLY (ABOUT 50
//  MINUTES OF ELAPSED TIME).  A 75TH ORDER SYSTEM PRODUCED AN
//  OVERFLOW CONDITION IN MATRIXX.  THE DISCRETE SYSTEM MODEL SHOULD
//  ALSO INCLUDE THE SAMPLING INTERVAL USED IN THE MODEL.  FOR SISO
//  SYSTEMS TO BE USED WITH TRANSFER FUNCTION REDUCTION METHODS, THE
//  NUMERATOR COEFFICIENTS, BI, AND THE DENOMINATOR COEFFICIENTS, AI,
//  MAY ALSO BE INCLUDED IN THE DATA FILE.  ANY OTHER DATA OR
//  VARIABLES INCLUDED IN THE DATA FILE WILL BE LOADED AND MAY CAUSE
//  UNEXPECTED AND/OR ERRATIC BEHAVIOR IN "MODRED".

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES:
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           3)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE:
//           1)  'FILENAME' - NAME OF DATA FILE CONTAINING SYST. MODEL
//                (N.B. - MUST CONTAIN SINGLE QUOTES.)
//                (N.B. - SEE DISCUSSION ABOVE)
//  DATA OUTPUT:
//       A)  TO STACK:  (ASSUMING PROPERLY CONSTRUCTED DATA FILE)
//           1)  F - F MATRIX OF STATE VARIABLE MODEL
//           2)  G - G MATRIX OF STATE VARIABLE MODEL
//           3)  H - H MATRIX OF STATE VARIABLE MODEL
//           4)  J - J MATRIX OF STATE VARIABLE MODEL
//           5)  TS - SAMPLING INTERVAL (IF DISCRETE-TIME MODEL)
//       B)  TO STORAGE FILE:
//           1)  F - F MATRIX OF STATE VARIABLE MODEL TO "SYSB"
//           2)  G - G MATRIX OF STATE VARIABLE MODEL TO "SYSB"
//           3)  H - H MATRIX OF STATE VARIABLE MODEL TO "SYSB"
//           4)  J - J MATRIX OF STATE VARIABLE MODEL TO "SYSB"
//           5)  TS - SAMPLING INT. (IF DISC.-TIME MODEL) TO "SYSB"
//       C)  TO CONSOLE:  (NONE)

//================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//================================================================

//  COMMAND FILE "LOAD" BEGINS EXECUTION HERE.

MSG=('  THIS BRANCH OF ""MODRED"" LOADS A SYSTEM MODEL.  THE MODEL  '
     '  IS LOADED ONTO THE STACK AND INTO A FILE NAMED ''SYSB''.    '
     '  ''SYSB'' IS USED BY METHODS USING STATE-VARIABLE MODELS.    '
```

```
        '  THE STATE-VARIABLE MODEL SHOULD DEFINE F,G,H,J, AND TS     '
        '  IN THE USUAL STATE-VARIABLE MODEL (TS IS THE SAMPLING      '
        '  INTERVAL, WHERE APPLICABLE).                               '
        '                                                             '
        '  ENTER THE NAME OF THE FILE CONTAINING THE SYSTEM MODEL.    '
        '  THE FILENAME MUST BE ENCLOSED IN APOSTROPHES.              '
        '  E.G. ''SYS1S''                                               ');
SKILMAC;

//  CLEAR STACK FOR LOADING NEW MODEL
CLEAR;

//  LOAD NEW MODEL
INQUIRE ANS 'ENTER FILENAME';
LOAD(ANS);
DISP('  --  COMPUTING, PLEASE WAIT  --')
SAVE('SYSB');
LOAD('SKILL')

DISP('  --  LOAD COMPLETED.  CARRIAGE RETURN TO CONTINUE--');
PAUSE;

//  RETURN CONTROL TO "MENU"

EXEC('MENU');

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "LOAD.DAT"

//#################################################################


$
```

```
//#################################################################

//  FILENAME: MACROS.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE:  6/24/85
//  VERSION 4.2 8/5/85

//  "MACROS" IS A COLLECTION OF PROMPTING MACROS THAT ARE USED
//  THROUGHOUT THE "MODRED" PROGRAM STRUCTURE.  THEY ARE STORED
//  IN THIS SEPARATE FILE TO FACILITATE RELOADING AFTER "LOAD"
//  HAS CLEARED THE STACK WHILE CREATING THE SYSTEM FILE "SYSB'.

//  DATA INPUT: (NONE)
//  DATA OUTPUT:
//       A)  TO STACK
//           1)  COMP - PROMPTING MESSAGE FOR COMPUTER DELAY
//           2)  COMPMAC - PROMPTING MACRO "      "      "
//           3)  SKILMAC - SUPPRESSES PROMPTING FOR EXPERTS
//           4)  WAIT - PROMPTING MESSAGE AFTER DISPLAYS
//           5)  WAITMAC - PROMPTING MACRO  "       "
//           6)  WHOMAC - DEBUGGING MACRO
//       B)  TO STORAGE FILES (NONE)
//       C)  TO CONSOLE (NONE)

//*****************************************************************

//     DEFINITION OF MACROS

//-----------------------------------------------------------------

//  MACRO "WAITMAC" BEGINS HERE.

//  "WAITMAC" PROVIDES FOR PROGRAM HALTS AT VARIOUS POINTS WHERE
//  THE OPERATOR MAY WANT TO STUDY THE DISPLAY.  FOR MOST USERS,
//  IT DISPLAYS THE PROMPT "WAIT" (BELOW).  FOR EXPERTS, IT
//  SUPPRESSES PROMPTS.

WAITMAC='IF SKIL3;DISP(WAIT);END;..
PAUSE;';

// END OF MACRO "WAITMAC".

//-----------------------------------------------------------------

//  MACRO "COMPMAC" BEGINS HERE.

//  "COMPMAC"  PROVIDES, FOR MOST USERS, THE PROMPT "COMP" BELOW
//  AFTER OPERATOR INPUT, AT POINTS IN THE PROGRAM WHERE THE
//  COMPUTING DELAY MIGHT BE MISCONSTRUED AS COMPUTER OR OPERATOR
//  ERROR.  FOR EXPERTS THE PROMPT IS SUPPRESSED.

COMPMAC='IF SKIL3;DISP(COMP);END;';

// END OF MACRO "COMPMAC".

//-----------------------------------------------------------------
```

```
//  MACRO "SKILMAC" BEGINS HERE.

//  "SKILMAC" IS USED TO SUPPRESS PROMPTING MESSAGES AND EXAMPLES FOR
//  USERS WHO ARE FAMILIAR WITH THE PROGRAM AND WISH TO PROGRESS
//  THROUGH THE PROGRAM A LITTLE FASTER.

SKILMAC='IF SKIL3;DISP(MSG);END;';

//  END OF MACRO "SKILMAC".

//-----------------------------------------------------------------

//  MACRO "WHOMAC" BEGINS HERE.

//  "WHOMAC" IS USED, DURING DEBUGGING TO PROVIDE ACCESS TO  A STACK
//  LIST AT SELECTED POINTS IN PROGRAM.  PROBABLY NOT USEFUL TO USERS.

WHOMAC='WHO;PAUSE;';

//  END OF MACRO "WHOMAC".

//-----------------------------------------------------------------

//     END OF MACRO DEFINITIONS

//*****************************************************************

// DEFINE STANDARD PROMPTING MESSAGES

COMP=(..
'                                                          ,
'    * * * * *    COMPUTING. PLEASE WAIT.    * * * * *'
'                                                        ');
WAIT=('   * * * * *    WAITING. CR TO CONTINUE.    * * * * *');

//-----------------------------------------------------------------

//  CREATE DATA FOR A BLANK SCREEN IMAGE.
//  USED IN PREPARING FULL-SCREEN MENUS, PROMPTS, ETC.

NULL=' ';
FOR I=1:25,NULVEC(I)=NULL;END;

//=================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=================================================================

//  COMMAND FILE "MACROS" BEGINS EXECUTION HERE.

//  RETURN CONTROL TO "MENU"

RETURN

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "MACROS.DAT"

//#################################################################
```

```
//#####################################################################

//  FILENAME: MENU.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE:  7/15/83
//  VERSION 4.2: 8/5/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED"
//  PROVIDES SELECTION OF MODEL REDUCTION METHOD

//  DATA INPUT:
//      A)  FROM OTHER PROGRAMS OR FILES
//              1)  COMPMAC - PROMPTING MACRO FROM "MODRED"
//              2)  NULVEC - DISPLAY VECTOR FROM "MODRED"
//              3)  SKIL - SELECTS PROMPTING LEVEL FROM "SKILL"
//  DATA OUTPUT: (NONE)

//====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//====================================================================

//  COMMAND FILE "MENU" BEGINS EXECUTION HERE.

CLEAR;
EXEC('MACROS');
DISP(NULVEC(1:6));
DISP('ENTER NUMBER OF DESIRED REDUCTION METHOD, OR 0 (TO EXIT).');
DISP ('      1 DOMINANT ROOT SELECTION          '
      '                                         '
      '      2 RESIDUE RETENTION                '
      '                                         '
      '      3 DIFFERENTIATION OF POLYNOMIALS   '
      '                                         '
      '      4 ""MOORE"" REDUCTION - CONTINUOUS '
      '                                         '
      '      5 ""MOORE"" REDUCTION - DISCRETE   '
      '                                         '
      '      6 SYSTEM MODEL INDEX               '
      '                                         '
      '      7 LOAD A SYSTEM MODEL              ');
LOAD('SKILL');
IF SKIL=2;INQUIRE ALS 'ENTER SELECTED OPTION NUMBER';END;
IF SKIL=3;INQUIRE ALS ' ';END;
IF ALS=0,RETURN,END;
COMPMAC;
MSG=('SELECTED OPTION NOT AVAILABLE.  TRY AGAIN.');
IF ALS7, DISP(MSG), EXEC('MENU');
IF ALS0, DISP(MSG), EXEC('MENU');
IF ALS=1; EXEC('RTSEL');
IF ALS=2; EXEC('RESREN');
IF ALS=3; EXEC('DIFPN');
IF ALS=4; EXEC('MOORE');
IF ALS=5; EXEC('DISCRT');
IF ALS=6; EXEC('INDEX');
IF ALS=7; EXEC('LOAD');

//  CONTROL RETURNS FROM SELECTED REDUCTION
```

```
//  RETURN CONTROL TO "MODRED"

RETURN;

//=============================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=============================================================================

//  END OF COMMAND FILE "MENU.DAT"

//#############################################################################

$
```

```
//

//  FILENAME: MODRED.DAT  --  "MODEL REDUCTION PROGRAM"
//  CREATED BY: K. R. DUNIPACE
//  DATE:  7/15/83
//  VERSION 4.2: 6/24/85
//  LAST UPDATED:  8/7/85

//  "MODRED" IS THE "TOP LEVEL" COMMAND FILE OF AN INTERACTIVE
//  MODEL REDUCTION PROGRAM IN MATRIXX LANGUAGE.  IT PROVIDES
//  INTRODUCTORY INFORMATION AND MACROS USED THROUGHOUT LOWER
//  LEVEL PROGRAMS.

//  DATA INPUT:
//      A)  FROM OTHER PROGRAMS AND FILES
//          1)  SKIL FROM FILE "SKILL"
//      B)  FROM CONSOLE
//          1)  SKIL - SELECTS LEVEL OF PROMPTING
//  DATA OUTPUT:
//      A)  TO STACK
//          1)  COMP - PROMPTING MESSAGE FOR COMPUTER DELAY
//          2)  COMPMAC - PROMPTING MACRO "      " "
//          3)  NULL -   BLANK SPACE FOR CREATING DISPLAYS
//          4)  NULVEC - BLANK VECTOR "      "      "
//          5)  SKIL - SELECTS LEVEL OF PROMPTING
//          6)  SKILMAC - SUPPRESSES PROMPTING FOR EXPERTS
//          7)  WAIT - PROMPTING MESSAGE AFTER DISPLAYS
//          8)  WAITMAC - PROMPTING MACRO  "      "
//          9)  WHOMAC - DEBUGGING MACRO
//      B)  TO STORAGE FILES (NONE)
//      C)  TO CONSOLE (NONE)

//*****************************************************************

//     DEFINITION OF MACROS

//---------------------------------------------------------------

//  MACRO "INSTR" BEGINS HERE.

//  "INSTR" WILL, EVENTUALLY, CONTAIN BRIEF INSTRUCTIONS FOR
//  USE OF "MODRED".

INSTR='DISP(..
''DETAILED INSTRUCTIONS ARE AVAILABLE IN THE USERS  GUIDE.  THESE '';..
''BRIEF INSTRUCTIONS ARE ONLY TO ASSIST IN STARTING THROUGH THE   '';..
''BUILT-IN DEMONSTRATIONS AND IN CHOOSING THE NOVICE/EXPERT       '';..
''OPTIONS FOR ROUTINE USE.  YOU HAVE CHOSEN THE INSTRUCTIONS      '';..
''THIS TIME.  WHEN YOU FINISH READING THESE INSTRUCTIONS AND      '';..
''CONTINUE INTO THE REDUCTION PROGRAMS (ENTER A CARRIAGE RETURN   '';..
''AFTER THE PAUSE PROMPT) THE PROGRAM WILL BE SET IN THE          '';..
''NOVICE MODE.  THE NOVICE MODE PROVIDES PROMPTING MESSAGES       '';..
''AND NUMERICAL EXAMPLES FOR A HANDS-ON DEMONSTRATION OF EACH     '';..
''SELECTED REDUCTION METHOD.  USERS WHO ARE FAMILIAR WITH THE     '';..
''PROGRAM MAY SELECT THE EXPERT MODE.  IN THIS MODE ALL THE       '';..
''EXAMPLES AND MOST OF THE PROMPTING MESSAGES ARE SUPPRESSED TO   '';..
''SPEED UP OPERATION AND SIMPLIFY THE OUTPUT DISPLAYS.            '';..
);WAITMAC;';
```

```
//  END OF MACRO "INSTR".

//-----------------------------------------------------------------

//     END OF MACRO DEFINITIONS

//*****************************************************************

//-----------------------------------------------------------------
//  CREATE DATA FOR A BLANK SCREEN IMAGE
//  USED IN PREPARING FULL-SCREEN PROMPTS, MENUS, ETC.
//  DOESN'T SEEM TO WORK AS EXPECTED.

NULL = ' ';
FOR I=1:25,NULVEC(I)=NULL;END;

//=================================================================
//     * * * * *     EXECUTION BEGINS  * * * * *
//=================================================================

//  COMMAND FILE "MODRED" BEGINS EXECUTION HERE.

//  LOAD PROMPTING MACROS

EXEC('MACROS')

//  RETURN FROM "MACROS"

DISP(" ";" ";" ";" ";" ";" ";" ";" ";" ";" ";" ";" ";" ";" ";);
DISP(..
'1234567890123456789012345678901234567890123456789012345678901234567890'
'2                                                                      2'
'3                                                                      3'
'4                                                                      4'
'5                                                                      5'
'6     ************************************************************     6'
'7     *                                                          *     7'
'8     *                        MODRED  V4.2                      *     8'
'9     *                            BY                            *     9'
'0     *                      K. R. DUNIPACE                      *     0'
'1     *    INDIANA UNIVERSITY-PURDUE UNIVERSITY-INDIANAPOLIS     *     1'
'2     *                          8/7/85                          *     2'
'3     *                                                          *     3'
'4     ************************************************************     4'
'5                                                                      5'
'6                                                                      6'
'7                          DEVELOPMENT OF                              7'
'8             THIS PUBLIC DOMAIN PROGRAM WAS SPONSORED BY              8'
'9               NASA - DRYDEN FLIGHT RESEARCH FACILITY                 9'
'0                                                                      0'
'1234567890123456789012345678901234567890123456789012345678901234567890'.

);
FOR I=1:15;NULVEC(I)=NULVEC(16-I);END;
DISP(" ";" ";" ";" ";" ";);
DISP('MODRED IS AN INTERACTIVE MODEL REDUCTION PROGRAM.'
      ,                                                 ,
      'THIS VERSION WORKS WITH MATRIXX VERSION 4.0.     '
      ,                                                 ,
      ,                                                 ,
```

```
      '    ENTER THE NUMBER OF THE DESIRED OPTION.      '
      '                                                 '
      '          1  INSTRUCTIONS                        '
      '          2  NOVICE                              '
      '          3  EXPERT                              ')
DISP(" ;" ;" ;" ;" ;");
INQUIRE SKIL 'ENTER DESIRED OPTION.';
//  DEFINE VARIABLE "SKIL" FOR LEVEL OF PROMPTING.
SAVE('SKILL',SKIL);
MSG=(' SELECTED OPTION NOT AVAILABLE, TRY AGAIN');
IF SKIL1,DISP(MSG),EXEC('MODRED');END;
IF SKIL3,DISP(MSG),EXEC('MODRED');END;

//  BRANCH TO MACRO "INSTR" FOR INSTRUCTIONS WITH OPTION 1.

IF SKIL=1;INSTR;SKIL=2;SAVE('SKILL',SKIL);END;

//  TRANSFER CONTROL TO COMMAND FILE "MENU"

EXEC('MENU');

//  RETURN FROM "MENU" AT END OF MODEL REDUCTION SESSION.

//  EXIT TO MATRIXX.

DISP('MODRED COMPLETED - CONTROL RETURNED TO MATRIXX');

//=================================================================
//     * * * * *     EXECUTION ENDS* * * * *
//=================================================================

//  END OF COMMAND FILE "MODRED.DAT"

//


$
```

```
//##################################################################

//  FILENAME:  MOORE.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE: 7/1/83
//  VERSION 4.2:  5/30/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED"
//  PERFORMS "SUBSYSTEM ELIMINATION" REDUCTION AS DESCRIBED BY B.C.
//  MOORE IN "PRINCIPAL COMPONENT ANALYSIS IN LINEAR SYSTEMS: ...",
//  IEEE AC-26, 1-81, PP 26FF.  THE ALGORITHM BELOW IMPLEMENTS THE
//  CONCEPT PRESENTED IN THE PAPER.  ALGORITHM WORKS FOR CONTINUOUS
//  SYSTEM (EITHER STABLE OR UNSTABLE) REPRESENTED BY THE STANDARD
//  STATE-VARIABLE MODEL:  XDOT = F X + G U; Y = H X + J U.
//  N.B. - "MOORE" BALANCES AND REDUCES SYSTEM MODEL ON BASIS OF
//  IMPULSE RESPONSE, BUT PRESENTS PLOTS COMPARING STEP RESPONSES
//  AND RMS ERROR BASED ON STEP RESPONSE.

//  DATA INPUTS:
//       A)  FROM OTHER PROGRAMS AND FILES:
//           1)  COMPMAC - PROMPTING MESSAGE FROM "MODRED"
//           2)  F - F MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           3)  FB - F MATRIX OF BALANCED SYSTEM FROM "BALANS"
//           4)  G - G MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           5)  GB - G MATRIX OF BALANCED SYSTEM FROM "BALANS"
//           6)  H - H MATRIX OF ORIG. SYSTEM FROM "SYSB"
//           7)  HB - H MATRIX OF BALANCED SYSTEM FROM "BALANS"
//           8)  J - J MATRIX OF ALL MODELS FROM "SYSB"
//           9)  NS - ORDER OF ORIGINAL SYSTEM FROM "BALANS"
//           10)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           11)  WAAITMAC - PROMPTING MESSAGE FROM "MODRED"
//       B)  FROM CONSOLE:
//           1)  NPTS - NO. OF POINTS FOR SIMULATION
//           2)  RCOL - SUBSYSTEM SELECTION VECTOR
//           3)  T - TIME DURATION FOR SIMULATION
//               (T/NPTS DETERMINES INCREMENT IN NUM. INTEG.)
//  DATA OUTPUT:
//       A)  TO STACK:
//           1)  FBR - F MATRIX OF REDUCED SYSTEM (BALANCED)
//           2)  GBR - G MATRIX OF REDUCED SYSTEM (BALANCED)
//           3)  HBR - H MATRIX OF REDUCED SYSTEM (BALANCED)
//               (F,G,H,J,FB,GB,HB,NS REMAIN ON STACK)
//       B)  TO STORAGE FILES:  (NONE)
//       C)  TO CONSOLE:
//           1)  ERR - RMS ERROR OF REDUCED STEP RESPONSES
//           2)  STR - STEP RESPONSES OF BOTH SYSTEMS AND ERROR

//================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//================================================================

//  COMMAND FILE "MOORE" BEGINS EXECUTION HERE.

//  BRANCH TO "BALANS" TO DETERMINE BALANCED MODEL

EXEC('BALANS');
```

```
//  RETURN FROM "BALANS"

//  SET UP PARAMETERS FOR SIMULATION
MSG=('ENTER TIME DURATION FOR SIMULATION.  E.G. 3.5');
SKILMAC;
INQUIRE T 'ENTER T';
MSG=('ENTER NUMBER OF POINTS FOR SIMULATION.  E.G. 20');
SKILMAC;
INQUIRE NPTS 'ENTER NPTS';
COMPMAC;
//  MAKE UP, FOR DISPLAY, STEP RESPONSES OF ORIGINAL SYSTEM.
//  FOLLOWS EXAMPLE IN MATRIXX USER'S GUIDE (VERSION 3.0 PG 4-28)
I,MI=SIZE(G);
HI,I=SIZE(H);
SE=[F,G,0*G;0*G',0*ONES(MI),DIAG(ONES(MI,1));H,J,0*ONES(HI,MI)];
SE=REAL(SE);
[TB,STRB]=TIMR(SE,NS+MI,[T,NPTS]);

//  NOTE:  MATRIXX AUTOMATICALLY GENERATES SYSTEM RESPONSE FOR
//  EACH INPUT/OUTPUT COMBINATION SEPARATELY.  HENCE, DIMENSION
//  OF [STRB] IS NPTS BY (HI * MI).

//  SELECT SUBSYSTEMS TO BE RETAINED
MSG=('SELECT COMPONENTS TO BE RETAINED.');
SKILMAC;
MSG=('ENTER, IN ORDER, ""0"" OR ""1"" FOR EACH COMPONENT.');
SKILMAC;
MSG=('""1"" RETAINS COMPONENT; ""0"" DELETES.');
SKILMAC;
MSG=('E.G. 1,1,0');
SKILMAC;
INQUIRE RCOL 'ENTER RCOL';
COMPMAC;
//  CREATE REDUCED SYSTEM MODEL BY SELECTING DESIRED SUBSYSTEMS FROM
//  BALANCED MODEL OF SYSTEM
CLEAR FBR GBR HBR;
NR=0;
FOR I=1:NS;..
IF RCOL(I)=1;NR=NR+1;GBR(NR,:)=GB(I,:);HBR(:,NR)=HB(:,I);....
NC=0;...
FOR IC=1:NS;IF RCOL(IC)=1;NC=NC+1;FBR(NR,NC)=FB(I,IC);END
NSR=SUM(RCOL);
//  MAKE UP, FOR DISPLAY, STEP RESPONSES OF REDUCED SYSTEM.
//  FOLLOWS EXAMPLE IN MATRIXX USER'S GUIDE (VERSION 3.0 PG 4-28)
SBR=[FBR,GBR,0*GBR;0*GBR',0*ONES(MI),DIAG(ONES(MI,1));HBR,...
      J,0*ONES(HI,MI)];
SBR=REAL(SBR);
[TBR,STBR]=TIMR(SBR,NSR+MI,[T,NPTS]);

//  SEE NOTE FOLL. [TB,STRB], ABOVE, DIM. [STBR] IS NPTS BY (RH*MI)

//  FORM AND PLOT STEP RESPONSES AND ERROR
STRE=STRB-STBR;
STR=[STRB,STBR,STRE];
ERR=0;
ERR=ERR+SUM(STRE.*STRE/SUM(STRB.*STRB));
CLEAR STRE STRB STBR;
RH,CH=SIZE(HBR);
MSG=('  --  WHEN FINISHED VIEWING PLOT, NOT NOW, CR TO CONT.  --');
SKILMAC;
```

```
FOR IC=1:RH*MI,PLOT(TB,STR(:,IC:RH*MI:3*RH*MI), ...
'YLAB/STEP RESPONSE/  XLAB/TIME SEC/...
 TITL/MODEL REDUCTION BY SUBSYSTEM ELIMINATION/TIC2 GRID=1 FULL');...
PAUSE;END;

// SEE NOTE FOLL. [TB,STRB], ABOVE, DIM. [STR] IS NPTS BY 3*(RH*MI).

PLOT('CLEAR')
//  CALCULATE AND DISPLAY RMS ERROR OF REDUCED RESPONSE
MSG=(' THE RMS ERROR FOR THIS REDUCTION IS:');
SKILMAC;
ERR=SQRT(ERR/(NPTS*RH*MI))

//  BRANCH TO APPROPRIATE MODULE FOR DISPLAY OF OUTPUT

IF RH+MI=2;EXEC('SING');ELSE EXEC('MULTI');END;END;

//  RETURN FROM OUTPUT MODULE

//  RETURN CONTROL TO "MENU"

EXEC('MENU');

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "MOORE.DAT"

//#################################################################


$
```

```
//#####################################################################

//  FILENAME:  MULTI.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE:  "SUMMER '83"
//  VERSION 4.2:  5/30/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  THIS EXEC FILE PRODUCES THE NON-GRAPHICAL OUTPUT RESULTS FOR
//  MULTIVARIABLE SYSTEMS.
//

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  FBR - F MATRIX OF BAL. SYST. FM "MOORE" OR "DISCRT"
//           2)  GBR - G MATRIX OF   "     "    "     "     "    "
//           3)  HBR - H MATRIX OF   "     "    "     "     "    "
//           4)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           5)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE (NONE)
//  DATA OUTPUT:
//       A)  TO STACK
//           1)  IGN - EIGENVALUES OF BALANCED REDUCED SYSTEM
//       B)  TO STORAGE (NONE)
//       C)  TO CONSOLE
//           1)  FBR - F MATRIX OF BALANCED REDUCED SYSTEM
//           2)  GBR - G   "     "    "         "       "
//           3)  HBR - H   "     "    "         "       "
//           4)  IGN - EIGENVALUES OF REDUCED SYSTEM

//====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//====================================================================

//  COMMAND FILE "MULTI" BEGINS EXECUTION HERE.

//  CALCULATE AND DISPLAY THE EIGENVALUES OF THE REDUCED SYSTEM
PO,D=EIG(FBR);
MSG=(' THE EIGENVALUES OF THE REDUCED SYSTEM ARE:');
SKILMAC
IGN=DIAG(D)
CLEAR PO D;
WAITMAC;
//  DISPLAY THE MATRICES OF THE REDUCED MODEL
MSG=(' IN THE REDUCED MODEL, THE COEFFICIENT MATRICES ARE:');
SHORT E;
SKILMAC;
FBR,GBR,HBR
SHORT
WAITMAC;

//  RETURN TO "MOORE" FOR CONT. SYSTS. OR "DISCRT" FOR DISC. SYSTS.

RETURN;

//====================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
```

```
//=====================================================================
//  END OF COMMAND FILE "MULTI.DAT"
//#####################################################################

$
```

```
//#####################################################################

//
//  FILENAME:  RESREN.DAT
//  CREATED BY: K. R. DUNIPACE
//  DATE:  6/27/83
//  VERSION 4.2:  5/30/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PROGRAM PERFORMS MODEL REDUCTION BY THE "RESIDUE RETENTION"
//  METHOD.  THE METHOD IS DESCRIBED IN "FLIGHT TEST TRAJECTORY
//  CONTROL ANALYSIS" BY R. WALKER AND N. GUPTA, INTEGRATED SYSTEMS
//  INC.  MAY 1982.
//  THIS METHOD IS VERY SIMILAR TO THE "DOMINANT ROOT" METHOD
//  PRESENTED IN THE PROGRAM "RTSEL".  THE DIFFERENCE BEING THAT THE
//  ZERO-FREQUENCY GAIN OF THE DISCARDED ROOTS IS ADDED TO THE
//  OUTPUT OF THE SYSTEM.
//

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  FD2 - DIAGONAL FORM OF ST.-VAR. MOD. FROM "SYSPRE"
//           3)  GD2 - DIAGONAL FORM OF ST.-VAR. MOD. FROM "SYSPRE"
//           4)  HD2 - DIAGONAL FORM OF ST.-VAR. MOD. FROM "SYSPRE"
//           5)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           6)  WAITMAC - PROMPTING MSG. FROM "MODRED".
//       B)  FROM CONSOLE
//           1)  NPTS - NUMBER OF POINTS IN SIMULATION FOR PLOTTING
//           2)  RCOL - VECTOR TO SELECT DOMINANT MODES.
//           3)  T - TIME DURATION FOR SIMULATION.
//  DATA OUTPUT:
//       A)  TO STACK:  (ALL VARIABLES - NONE CLEARED)
//       B)  TO STORAGE FILES:  (NONE)
//       C)  TO CONSOLE:
//           1)  ERR - RMS ERROR OF REDUCED STEP RESP.
//           2)  FR - "F" MATRIX OF REDUCED ST.-VAR. MODEL
//           3)  G - COEFFS. OF NUMERATOR OF REDUCED TRANS. FN.
//           4)  GR - "G" MATRIX OF REDUCED ST.-VAR. MODEL
//           5)  HR - "H" MATRIX OF REDUCED ST.-VAR. MODEL
//           6)  PDEN - COEFFS. OF DENOMINATOR OF RED. TRANS. FN.
//           7)  STR - STEP RESPONSE OF BOTH SYSTEMS AND ERROR

//====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//====================================================================

//  COMMAND FILE "RESREN" BEGINS EXECUTION HERE.

//  BRANCH TO "SYSPRE" TO SET UP STATE-VARIABLE MODEL.

EXEC('SYSPRE');

//  RETURN FROM "SYSPRE"

MSG=(' SELECT RESIDUES TO BE RETAINED.  ENTER, IN ORDER, 0 OR 1 FOR  ');
SKILMAC;
```

```
MSG=(' EACH RESIDUE.  ""1"" RETAINS RESIDUE, ""0"" DELETES.');
SKILMAC;
MSG=(' E.G. 0,1,1,1');
SKILMAC;
INQUIRE RCOL 'ENTER RCOL';
COMPMAC;
CLEAR FR GR HR;
NNR=0;
NR=0;
//  ASSIGN ELEMENTS OF F,G, AND H MATRICES OF STATE VARIABLE MODEL TO
//  EITHER REDUCED STATE VARIABLE MODEL OR TO TEMPORARY VECTORS FOR
//  THE "RESIDUE RETENTION"  CALCULATION.
FOR I=1:NS+1,...
     IF RCOL(I)=1;NR=NR+1;GR(NR)=GD2(I);HR(1,NR)=HD2(1,I);...
     FR(NR,NR)=FD2(I,I);...
 ELSE NNR=NNR+1;GN(NNR)=GD2(I);FN(NNR,NNR)=FD2(NNR,NNR);...
 HN(1,NNR)=HD2(1,I);END;
//  PERFORM "RESIDUE RETENTION"CALCULATIONS
JN=J;
FOR K=1:NNR,JN=JN+HN(K)*GN(K)/ABS(FN(K,K));
HR(1,NR)=HR(1,NR)+JN;
//  CONVERT DIAGONAL STATE-VARIABLE MODEL TO "JORDAN FORM" FOR
//  COMPLEX EIGENVALUES.
NSR=SUM(RCOL);
I=SQRT(-1);
DVEC=DIAG(FR);
[NF,RI]=SIZE(DVEC);
P3=0*ONES(NF)+EYE(NF);
FOR K=1:NF-1,...
    IF ABS(DVEC(K)-CONJG(DVEC(K+1)))1.E-10;P3(K,K)=1/2;...
 P3(K+1,K+1)=I/2;P3(K,K+1)=-I/2;P3(K+1,K)=1/2;END
FR=P3\FR*P3;
MSG=('THE REDUCED-AUGMENTED MODEL IS:');
SKILMAC;
FR=REAL(FR)
GR=P3\GR;
GR=REAL(GR)
HR=HR*P3;
HR=REAL(HR)
WAITMAC
//  CONSTRUCT "SYSTEM" MODEL FOR MATRIXX SIMULATION OF STEP RESPONSE
SE=[F,G,0*G;0*G',0,1;H,J,0];
SE=REAL(SE);
MSG=(' ENTER TIME DURATION FOR SIMULATION.  E.G. 3.5');
SKILMAC;
INQUIRE T 'ENTER T';
MSG=(' ENTER NUMBER OF POINTS FOR SIMULATION.  E.G. 20');
SKILMAC;
INQUIRE NPTS 'ENTER NPTS';
COMPMAC
//  PREPARE STEP RESPONSE SIMULATION FOR PLOTTING
[TD,STRF]=TIMR(SE,NS+1,[T,NPTS]);
SER=[FR,GR;HR,0];
SER=REAL(SER);
[TR,STRR]=TIMR(SER,NSR,[T,NPTS]);
STRE=STRF-STRR;
ERR=0;
ERR=ERR+SUM(STRE.*STRE/SUM(STRF.*STRF));
STR=[STRF,STRR,STRE];
CLEAR STRE;STRF;STRR;
```

```
MSG=('  --  WHEN FINISHED VIEWING PLOT, NOT NOW, CR TO CONT.  --  ');
SKILMAC;
PLOT(TD,STR,'YLAB /STEP RESPONSE/ XLAB/TIME [SEC]/ TITL/MODEL ...
REDUCTION BY RESIDUE RETENTION/ TIC2 FULL ');
PAUSE;
PLOT('CLEAR');
MSG=(' THE RMS ERROR FOR THIS REDUCTION IS:');
SKILMAC;
ERR=SQRT(ERR/NPTS)
//  CALCULATE TRANSFER FUNCTION FROM REDUCED STATE-VARIABLE MODEL,
//  (INCLUDING "RETAINED RESIDUES").
MSG=(' IN THE REDUCED TRANSFER FUNCTION,');
SKILMAC;
PDEN=POLY(DVEC);
K,I=SIZE(PDEN);
FOB=DIAG(ONES(K-2,1),1);
FOB(:,1)=-PDEN(2:K);
PO,D=EIG(FOB);
MSG=(' THE COEFFICIENTS OF THE NUMERATOR ARE:');
SKILMAC;
G=PO*(HR./CONJG(PO(1,:)))'
MSG=(' THE COEFFICIENTS OF THE DENOMINATOR ARE:');
SKILMAC;
PDEN(1:K-1)
WAITMAC;

//  RETURN CONTROL TO "MENU"

EXEC('MENU')

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "RESREN.DAT"

//#################################################################


$
```

```
//#################################################################

//
//  FILENAME:  RTSEL.DAT
//  CREATED: BY K. R. DUNIPACE
//  DATE:  7/14/83
//  VERSION 4.2:  5/30/85

//  PART OF MODEL REDUCTION PORGRAM "MODRED".
//  "RTSEL" PERFORMS MODEL REDUCTION OF SINGLE-INPUT SINGLE-OUTPUT
//  CONTINUOUS SYSTEMS MODELED BY TRANSFER FUNCTIONS.  REDUCTION IS
//  ACHIEVED INTERACTIVELY BY SELECTING (AND REMOVING) RESIDUES AND
//  ASSOCIATED EIGENVALUES WHICH ARE CONSIDERED BY THE DESIGNER TO
//  BE NEGLIGIBLE IN COMPARISON TO RESIDUES OF DOMINANT ROOTS.
//  RESIDUES ARE FOR STEP RESPONSE.

//  DATA INPUTS:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  FD2 - DIAG. FORM OF ST.-VAR. MODEL; FROM "SYSPRE"
//           3)  GD2 - G MATRIX FOR DIAGONAL MODEL;  FROM "SYSPRE"
//           4)  HD2 - H MATRIX FOR DIAGONAL MODEL;  FROM "SYSPRE"
//           5)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           6)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE
//           1)  NPTS - NUMBER OF POINTS IN SIMULATION FOR PLOTTING
//           2)  RCOL - VECTOR TO SELECT DOMINANT ROOTS
//           3)  T - TIME DURATION FOR SIMULATION
//  DATA OUTPUT:
//       A)  TO STACK:  (ALL VARIABLES - NONE CLEARED)
//       B)  TO STORAGE FILES:  (NONE)
//       C)  TO CONSOLE:
//           1)  ERR - RMS ERROR OF REDUCED STEP RESPONSE
//           2)  FR - "F" MATRIX OF RED. ST.-VAR. MODEL
//           3)  GR - "G" MATRIX OF RED. ST.-VAR. MODEL
//           4)  HR - "H" MATRIX OF RED. ST.-VAR. MODEL
//           5)  NUM - COEFFS. OF NUMERATOR OF REDUCED TRANS. FN.
//           6)  PDEN - COEFFS. OF DENOMINATOR OF RED. TRANS. FN.
//           7)  STR - STEP RESPONSE OF BOTH SYSTEMS AND ERROR

//================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//================================================================

//  COMMAND FILE "RTSEL" BEGINS EXECUTION HERE.

//(ALL ABOVE INPUTS NORMALLY PROVIDED AUTOMATICALLY BY ""SYSPRE"");
//T AND NPTS WILL BE ENTERED MANUALLY.

EXEC('SYSPRE');
MSG=('SELECT RESIDUES TO BE RETAINED.');
SKILMAC;
MSG=('ENTER, IN ORDER, 0 OR 1 FOR EACH RESIDUE.');
SKILMAC;
MSG=('""1"" RETAINS RESIDUE; MUST RETAIN LAST RESIDUE.');
SKILMAC;
MSG=('E.G.: 0,1,1,1');
```

```
SKILMAC;
INQUIRE RCOL 'ENTER RCOL';
COMPMAC;
NR=0;
CLEAR FR GR HR;
FOR RI=1:NS+1;...
     IF RCOL(RI)=1;NR=NR+1;GR(NR)=GD2(RI);HR(1,NR)=HD2(1,RI);...
     NC=0;...
 FOR CI=1:NS+1;...
     IF RCOL(CI)=1;NC=NC+1;FR(NR,NC)=FD2(RI,CI);END
NSR=SUM(RCOL);
I=SQRT(-1);
DVEC=DIAG(FR);
[NF,RI]=SIZE(DVEC);
P3=0*ONES(NF)+EYE(NF);
FOR K=1:NF-1,...
    IF ABS(DVEC(K)-CONJG(DVEC(K+1)))1.E-10;P3(K,K)=1/2;...
 P3(K+1,K+1)=I/2;...
 P3(K,K+1)=-I/2;P3(K+1,K)=1/2;END
FR=P3\FR*P3;
MSG=('THE REDUCED/AUGMENTED MODEL IS:');
SKILMAC;
FR=REAL(FR)
GR=P3\GR;
GR=REAL(GR)
HR=HR*P3;
HR=REAL(HR)
WAITMAC;
SE=[F,G,0*G;0*G',0,1;H,J,0];
MSG=('ENTER TIME DURATION FOR SIMULATION.  E.G. 3.5');
SKILMAC;
INQUIRE T 'ENTER T';
MSG=('ENTER NUMBER OF POINTS FOR SIMULATION.  E.G. 20');
SKILMAC;
INQUIRE NPTS 'ENTER NPTS';
COMPMAC;
SE=REAL(SE);
[TD,STRF]=TIMR(SE,NS+1,[T,NPTS]);
SER=[FR,GR;HR,0];
SER=REAL(SER);
[TR,STRR]=TIMR(SER,NSR,[T,NPTS]);
STRE=STRF-STRR;
STR=[STRF,STRR,STRE];
ERR=0;
ERR=ERR+SUM(STRE.*STRE/SUM(STRF.*STRF));
CLEAR STRE STRF STRR;
MSG=('  --  WHEN FINISHED VIEWING PLOT, NOT NOW, CR TO CONT.  --');
SKILMAC;
PLOT(TR,STR,'YLAB/ STEP RESPONSE/XLAB/TIME [SEC]/ TITL/MODEL ...
REDUCTION BY DOMINANT RESIDUE SELECTION/ TIC2 FULL ')
PAUSE;
PLOT('CLEAR')
MSG=('THE RMS ERROR FOR THIS REDUCTION IS:');
SKILMAC;
ERR=SQRT(ERR/NPTS)
MSG=(' IN THE REDUCED TRANSFER FUNCTION,');
SKILMAC;
PDEN=POLY(DVEC);
K,I=SIZE(PDEN);
FOB=DIAG(ONES(K-2,1),1);
```

```
FOB(:,1)=-PDEN(2:K);
PO,D=EIG(FOB);
MSG=(' THE COEFFICIENTS OF THE NUMERATOR ARE:');
SKILMAC;
NUM=PO*(HR./CONJG(PO(1,:)))'
MSG=(' THE COEFFICIENTS OF THE DENOMINATOR ARE:');
SKILMAC;
DEN=PDEN(1:K-1)
WAITMAC;

//  RETURN CONTROL TO "MENU"

EXEC('MENU');

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "RTSEL.DAT"

//#################################################################


$
```

```
//#####################################################################

//  FILENAME:  SING.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE: SUMMER '83
//  VERSION 4.2:  3/30/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  DISPLAYS OUTPUT INFORMATION FOR SINGLE-INPUT SINGLE-OUTPUT
//  SYSTEMS IN TRANSFER FUNCTION FORM.

//  DATA INPUT
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  FBR - F MAT. - BAL. RED. SYST. - "DISCRT" OR "MOORE"
//           2)  GBR - G MAT.     "    "      "          "     "    "
//           3)  HBR - H MAT.     "    "      "          "     "    "
//           4)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           5)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE (NONE)
//  DATA OUTPUT
//       A)  TO STACK
//           1)  GO - NUMERATOR COEFFICIENTS OF REDUCED TRANSFER FN.
//           2)  IGNOI - EIGENVALUES OF REDUCED SYSTEM
//           3)  NS - ORDER OF REDUCED MODEL
//           4)  PDEN - DENOMINATOR COEFF. OF REDUCED TRANS. FN.
//           5)  RESI - RESIDUES OF REDUCED TRANS. FN.
//       B)  TO STORAGE (NONE)
//       C)  TO CONSOLE
//           1)  GO - NUM. COEFF. OF REDUCED TRANS. FN.
//           2)  IGNOI - EIGENVALUES OF REDUCED SYSTEM
//           3)  PDEN - DEN. COEFF. OF REDUCED TRANS. FN.
//           4)  RESI - RESIDUES OF REDUCED TRANS. FN.

//=====================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//=====================================================================

//  COMMAND FILE "SING" BEGINS EXECUTION HERE.

//  REVERSE DIAGONALIZING PROCESS TO CONVERT DIAGONAL MATRIX BACK TO
//  "OBSERVER CANONICAL FORM" SO THAT COEFFICIENTS OF TRANSFER
//  FUNCTION CAN BE EXTRACTED.
[NS,DUM]=SIZE(FBR);
[P1,LAMB]=EIG(FBR);
P1NGB=INV(P1)*GBR;
P2=DIAG(P1NGB);
RESB=HBR*P1*P2;
IGNB=DIAG(LAMB);
INDX=SORT(IGNB);
FOR I=1:NS,...
      IGNBI(I)=IGNB(INDX(I));...
      RESI(I)=RESB(INDX(I));END;
LAMBI=DIAG(IGNBI);
PDEN=POLY(DIAG(LAMBI));
IF NS1;FO=DIAG(ONES(NS-1,1),1);FO(:,1)=-PDEN(2:NS+1);...
    ELSE FO=-PDEN(2);END;
[P1O,LAM]=EIG(FO);
```

```
IGNO=DIAG(LAM);
INDXO=SORT(IGNO);
FOR I=1:NS,...
     IGNOI(I)=IGNO(INDXO(I));...
     P1OI(:,I)=P1O(:,INDXO(I));END;
LAMI=DIAG(IGNOI);
P2O=RESI./CONJG((P1OI(1,:))');
GO=P1OI*P2O;
MSG=(' THE EIGENVALUES OF THE REDUCED SYSTEM ARE:');
SKILMAC;
IGNOI
MSG=(' THE RESIDUES OF THE REDUCED SYSTEM ARE:');
SKILMAC;
RESI
WAITMAC;
MSG=(' IN THE REDUCED TRANSFER FUNCTION, THE COEFFICIENTS OF '
      ' THE NUMERATOR POLYNOMIAL ARE (NUM. PDEN MAY NOT BE    '
      ' CORRECT IF RESIDUES ARE COMPLEX.):                     ');
SKILMAC;
GO
MSG=(' THE COEFFICIENTS OF THE DENOMINATOR POLYNOMIAL ARE:');
SKILMAC;
PDEN
WAITMAC;
CLEAR DUM P1 LAMB P1NGB P2 RESB IGNB INDX IGNBI LAMBI FO P1O;
CLEAR LAM IGNO INDXO P1OI LAMI P2O;

//  RETURN TO "MOORE" FOR CONT. SYSTS. OR "DISCRT" FOR DIS. SYSTS.

RETURN;

//=================================================================
//     * * * * *     EXECUTION ENDS     * * * * *
//=================================================================

//  END OF COMMAND FILE "SING.DAT"

//#################################################################


$
```

```
//#################################################################

//  FILENAME:  SYSPRE.DAT
//  CREATED BY:  K. R. DUNIPACE
//  DATE:  SUMMER '83
//  VERSION 4.2:  5/30/85

//  PART OF MODEL REDUCTION PROGRAM "MODRED".
//  PREPARES SYSTEM MODEL FROM TRANSFER FUNCTION COEFFICIENTS.
//  MODEL IS DIAGONAL STATE VARIABLE MODEL WITH UNIT VALUES IN
//  ALL ELEMENTS OF THE "G" MATRIX.  THUS, THE ELEMENTS IN THE
//  "H" MATRIX ARE THE RESIDUES OF THE SYSTEM IMPULSE RESPONSE.
//  IN THIS APPLICATION, THE USUAL SYSTEM MODEL IS AUGMENTED SO
//  THE IMPULSE RESPONSE OF THE AUGMENTED MODEL IS THE STEP
//  RESPONSE OF THE TRANSFER FUNCTION.

//  DATA INPUT:
//       A)  FROM OTHER PROGRAMS AND FILES
//           1)  COMPMAC - PROMPTING MSG. FROM "MODRED"
//           2)  SKILMAC - PROMPT SUPPRESSION MACRO FROM "MODRED"
//           3)  WAITMAC - PROMPTING MSG. FROM "MODRED"
//       B)  FROM CONSOLE
//           1)  AI - DENOMINATOR COEFFICIENTS OF TRANS. FN.
//           2)  BI - NUMERATOR COEFFICIENTS OF TRANS. FN.
//  DATA OUTPUT:
//       A)  TO STACK:  (ALL VARIABLES - NONE CLEARED)
//       B)  TO STORAGE FILES:  (NONE)
//       C)  TO CONSOLE:
//           1)  IGN - EIGENVALUES OF AUGMENTED SYSTEM MODEL
//           2)  RES - RESIDUES OF STEP RESPONSE OF TRANS. FN.

//================================================================
//     * * * * *     EXECUTION BEGINS     * * * * *
//================================================================

//  COMMAND FILE "SYSPRE" BEGINS EXECUTION HERE.

MSG=('ENTER COEFFICIENTS OF NUMERATOR POLYNOMIAL.');
SKILMAC;
MSG=('E.G. [0.48,4.8,12]');
SKILMAC;
INQUIRE BI 'ENTER BI';
MSG=('ENTER COEFFICIENTS OF DENOMINATOR POLYNOMIAL.');
SKILMAC;
MSG=('E.G. [9,20,12]');
SKILMAC;
MSG=('N.B. COEFF. OF ""S**N"" (NOT ENTERED) MUST BE 1.');
SKILMAC;
INQUIRE AI 'ENTER AI';
//  FORM STATE VARIABLE MODEL IN "OBSERVER CANONICAL" FORM.
I,NS=SIZE(AI);
I,MI=SIZE(BI);
C=0;
ON1=ONES(NS-1,1);
F=DIAG(ON1,1);
F(:,1)=-AI(1:NS)';
G=0*ONES(1,NS-MI),BI';
```

```
H=[1,0*ON1'];
J=C;
//  AUGMENT STATE-VARIABLE MODEL FOR OBTAINING STEP-RESPONSE
FA=[F,G;0*G',0];
GA=[0*G;1];
HA=[H,J];
//  DIAGONALIZE MODEL
P1,D=EIG(FA);
GD=INV(P1)*GA;
HD=HA*P1;
//  TRANSFORM DIAGONAL MODEL WITH DIAGONAL SIMILARITY TRANSFORM
//  SO ALL ELEMENTS OF "G" MATRIX ARE 1.0
P2=DIAG(GD);
FD2=INV(P2)*D*P2;
//  DISPLAY EIGENVALUES AND RESIDUES
MSG=('THE EIGENVALUES OF THE AUGMENTED SYSTEM ARE:');
SKILMAC;
IGN=DIAG(D)
WAITMAC;
MSG=('THE RESIDUES OF THE AUGMENTED SYSTEM ARE:');
SKILMAC;
HD2=HD*P2;
RES=HD2'
WAITMAC;
GD2=P2\GD;

//  RETURN CONTROL TO "RTSEL" OR "RESREN"

RETURN;

//=================================================================
//      * * * * *      EXECUTION ENDS      * * * * *
//=================================================================

//  END OF COMMAND FILE "SYSPRE.DAT"

//#################################################################


$
```

MODRED USER'S GUIDE

Appendix C

System Models

Demonstration System #1

Transfer Function Model

$$T(s) = \frac{0.48S^2 + 4.8S + 12}{S^3 + 9S^2 + 20S + 12}$$

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -9 & 1 & 0 \\ -20 & 0 & 1 \\ -12 & 0 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0.48 \\ 4.8 \\ 12 \end{bmatrix} u(t)$$

$$\underline{y}(t) = [1\ 0\ 0\ ]\underline{x}(t) + 0\ u(t)$$

Discrete State Variable Model

$$\underline{x}(k+1) = \begin{bmatrix} 0.1149 & 0.0786 & 0.0086 \\ -1.6748 & 0.8224 & 0.1556 \\ -0.9433 & -0.1027 & 0.9935 \end{bmatrix} \underline{x}(k) + \begin{bmatrix} 0.0853 \\ 0.8225 \\ 1.9163 \end{bmatrix} u(k)$$

$$\underline{y}(k) = [1\ 0\ 0\ ]\underline{x}(k) + 0\ u(k)$$

(N. B. $T_s = 0.1667$ secs.)

Demonstration system #1 is a single-input single-output system with three real roots. One of the roots is strongly non-dominant. A transfer function model (SYS1), a continuous state-variable model (SYS1S), and a discrete-time state-variable model (SYS1D) are available.

Demonstration System #2A

Transfer Function Model

$$T(s) = \frac{2.5S^2 + 5.625S + 2.8125}{S^3 + 4.5S^2 + 6.3125S + 2.8125}$$

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -4.5 & 1 & 0 \\ -6.3125 & 0 & 1 \\ -2.8125 & 0 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 2.5 \\ 5.625 \\ 2.8125 \end{bmatrix} u(t)$$

$$\underline{y}(t) = [1\ 0\ 0\ ]\underline{x}(t) + 0\ u(t)$$

Discrete State Variable Model

$$\underline{x}(k+1) = \begin{bmatrix} 0.2272 & 0.1403 & 0.0215 \\ -0.9462 & 0.8586 & 0.2372 \\ -0.3946 & -0.0605 & 0.9944 \end{bmatrix} \underline{x}(k) + \begin{bmatrix} 0.4774 \\ 1.0660 \\ 0.5195 \end{bmatrix} u(k)$$

$$\underline{y}(k) = [1\ 0\ 0\ ]\underline{x}(k) + 0\ u(k)$$

(N. B. $T_s = 0.25$ secs.)

Demonstration system #2A is a single-input single-output system with three real roots. All of the roots contribute equally to the response. A transfer function model (SYS2A), a continuous state-variable model (SYS2AS), and a discrete-time state-variable model (SYS2AD) are available.

Demonstration System #3A

Transfer Function Model

$$T(s) = \frac{6.6667S^2 + 40S + 60}{S^3 + 12S^2 + 46S + 60}$$

Continuous State Variable Model

$$\underline{\dot{x}(t)} = \begin{bmatrix} -12 & 1 & 0 \\ -46 & 0 & 1 \\ -60 & 0 & 0 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 6.6667 \\ 40 \\ 60 \end{bmatrix} u(t)$$

$$\underline{y(t)} = [1\ 0\ 0\ ]\underline{x(t)} + 0\ u(t)$$

Demonstration system #3A is a single-input single-output system with two complex roots and a real root. The real root is strongly dominant. A transfer function model (SYS3A) and a continuous state-variable model (SYS3AS) are available.

Demonstration System #3B

Transfer Function Model

$$T[s] = \frac{1.1111S^2 + 1.1111S + 0.278}{S^3 + 2S^2 + 1.2778S + 0.2778}$$

Continuous State Variable Model

$$\underline{\dot{x}[t]} = \begin{bmatrix} -2 & 1 & 0 \\ -1.2778 & 0 & 1 \\ -0.2778 & 0 & 0 \end{bmatrix} \underline{x[t]} + \begin{bmatrix} 1.1111 \\ 1.1111 \\ 0.2778 \end{bmatrix} u[t]$$

$$\underline{y[t]} = [1\ 0\ 0\ ]\underline{x[t]} + 0\ u[t]$$

Discrete State Variable Model

$$\underline{x[k+1]} = \begin{bmatrix} 0.6890 & 0.1467 & 0.0136 \\ -0.1912 & 0.9824 & 0.1739 \\ -0.0407 & -0.0038 & 0.9998 \end{bmatrix} \underline{x[k]} + \begin{bmatrix} 0.1784 \\ 0.1779 \\ 0.0441 \end{bmatrix} u[k]$$

$$\underline{y[k]} = [1\ 0\ 0\ ]\underline{x[k]} + 0\ u[k]$$

[N. B. T_s = 0.175 secs.]

Demonstration system #3B is a single-input single-output system with two complex roots and one real root. The real root is strongly dominant. It is similar to system #3A except that it is time-scaled to have about the same settling time as system #1. A transfer function model (SYS3B), a continuous state-variable model (SYS3BS), and a discrete-time state-variable model (SYS3BD) are available.

Demonstration System #4

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -9 & 1 & 0 \\ -20 & 0 & 1 \\ -12 & 0 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0.48 & 6 \\ 4.8 & 2 \\ 12 & 12 \end{bmatrix} u(t)$$

$$\underline{y}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 2.28 & 0.376 & 1.078 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u}(t)$$

Demonstration system #4 is a multi-input multi-output system with three real roots. It is a modification of sstem #1 to include a second input and a second output. Thus, one of the roots is strongly non-dominant. The second output is much larger than the first. A continuous state-variable model (SYS4S) is available.

Demonstration System #4A

Continuous State Variable Model

$$\underline{\dot{x}(t)} = \begin{bmatrix} -9 & 1 & 0 \\ -20 & 0 & 1 \\ -12 & 0 & 0 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0.48 & 6 \\ 4.8 & 2 \\ 12 & 12 \end{bmatrix} u(t)$$

$$\underline{y(t)} = \begin{bmatrix} 1 & 0 & 0 \\ 0.8946 & 0.1440 & 0.4230 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u(t)}$$

Discrete State Variable Model

$$\underline{x[k+1]} = \begin{bmatrix} 0.1149 & 0.0786 & 0.0086 \\ -1.6748 & 0.8224 & 0.1556 \\ -0.9433 & -0.1027 & 0.9935 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.0853 & 0.4952 \\ 0.8225 & -0.5938 \\ 1.9163 & 1.3677 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = \begin{bmatrix} 1 & 0 & 0 \\ 0.8946 & 0.1440 & 0.4230 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u(k)}$$

(N. B.: T_s = 0.1667 secs.)

Demonstration system #4A is a multi-input multi-output system with three real roots. It is system #4 modified so that both outputs are about equal. A continuous state-variable model (SYS4AS), and a discrete-time state-variable model (SYS4AD) are available.

Demonstration System #5

Transfer Function Model

$$T(s) = \frac{-0.48S^2 + -4.8S + -12}{S^3 + 7S^2 + 4S + -12}$$

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -7 & 1 & 0 \\ -4.8 & 0 & 1 \\ 12 & 0 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} -0.48 \\ -4.8 \\ -12 \end{bmatrix} u(t)$$

$$\underline{y}(t) = [1\ 0\ 0\ ]\underline{x}(t) + 0\ u(t)$$

Discrete State Variable Model

$$\underline{x}(k+1) = \begin{bmatrix} 0.2904 & 0.0968 & 0.0097 \\ -0.2712 & 0.9683 & 0.1646 \\ 1.1622 & 0.1162 & 1.0071 \end{bmatrix} \underline{x}(k) + \begin{bmatrix} -0.1000 \\ -0.9406 \\ -2.0933 \end{bmatrix} u(k)$$

$$\underline{y}(k) = [1\ 0\ 0\ ]\underline{x}(k) + 0\ u(k)$$

(N. B. T_s = 0.1667 secs.)

Demonstration system #5 is a single-input single-output system with three real roots. One of the roots is unstable. A transfer function model (SYS5), a continuous state-variable model (SYS5S), and a discrete-time state-variable model (SYS5D) are available.

Demonstration System #5

Transfer Function Model

$$T[s] = \frac{-0.48S^2 + -4.8S + -12}{S^3 + 7S^2 + 4S + -12}$$

Continuous State Variable Model

$$\dot{\underline{x}}[t] = \begin{bmatrix} -7 & 1 & 0 \\ -4 & 0 & 1 \\ 12 & 0 & 0 \end{bmatrix} \underline{x}[t] + \begin{bmatrix} -0.48 \\ -4.8 \\ -12 \end{bmatrix} u[t]$$

$$\underline{y}[t] = [1\ 0\ 0\ ]\underline{x}[t] + 0\ u[t]$$

Discrete State Variable Model

$$\underline{x}[k+1] = \begin{bmatrix} 0.2904 & 0.0968 & 0.0097 \\ -0.2712 & 0.9683 & 0.1646 \\ 1.1622 & 0.1162 & 1.0071 \end{bmatrix} \underline{x}[k] + \begin{bmatrix} -0.1000 \\ -0.9406 \\ -2.0933 \end{bmatrix} u[k]$$

$$\underline{y}[k] = [1\ 0\ 0\ ]\underline{x}[k] + 0\ u[k]$$

(N. B. $T_s = 0.1667$ secs.)

Demonstration system #5 is a single-input single-output system with three real roots. One of the roots is unstable. A transfer function model (SYS5), a continuous state-variable model (SYS5S), and a discrete-time state-variable model (SYS5D) are available.

Demonstration System #6

Continuous State Variable Model

$$\underline{\dot{x}(t)} = \begin{bmatrix} -7 & 1 & 0 \\ -4 & 0 & 1 \\ 12 & 0 & 0 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0.48 & 6 \\ 4.8 & 2 \\ 12 & 12 \end{bmatrix} \underline{u(t)}$$

$$\underline{y(t)} = [1\ 0\ 0\ ]\underline{x(t)} + 0\ u(t)$$

Discrete State Variable Model

$$\underline{x(k+1)} = \begin{bmatrix} 0.2717 & 0.0992 & 0.0105 \\ -0.2708 & 0.9661 & 0.1727 \\ 1.1902 & 0.1260 & 1.0081 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.1061 & 0.6242 \\ 0.9950 & 0.3242 \\ 2.2036 & 2.8766 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = [1\ 0\ 0\ ]\underline{x(k)} + 0\ u(k)$$

(N. B. T_s = 0.175 secs.)

Demonstration system #6 is a multi-input single-output system with three real roots. One of the roots is unstable. A continuous state-variable model (SYS6S), and a discrete-time state-variable model (SYS6D) are available.

Demonstration System #7

Continuous State Variable Model

$$\underline{\dot{x}(t)} = \begin{bmatrix} -7 & 1 & 0 \\ -4 & 0 & 1 \\ 12 & 0 & 0 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0.48 & 6 \\ 4.8 & 2 \\ 12 & 12 \end{bmatrix} \underline{u(t)}$$

$$\underline{y(t)} = \begin{bmatrix} 1 & 0 & 0 \\ 0.8946 & 0.1440 & 0.4230 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u(t)}$$

Discrete State Variable Model

$$\underline{x(k+1)} = \begin{bmatrix} 0.2717 & 0.0992 & 0.0105 \\ -0.2708 & 0.9661 & 0.1727 \\ 1.1902 & 0.1260 & 1.0081 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0.1061 & 0.6242 \\ 0.9950 & 0.3242 \\ 2.2036 & 2.8766 \end{bmatrix} \underline{u(k)}$$

$$\underline{y(k)} = \begin{bmatrix} 1 & 0 & 0 \\ 0.8946 & 0.1440 & 0.4230 \end{bmatrix} \underline{x(k)} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u(k)}$$

(N. B. T_s = 0.175 secs.)

Demonstration system #7 is a multi-input multi-output system with three real roots. One of the roots is unstable. It is a modification of system #6 to include a second output. A continuous state-variable model (SYS7S), and a discrete-time state-variable model (SYS7D) are available.

Demonstration System #9

Discrete State Variable Model

$$\underline{x[k+1]} = \begin{bmatrix} 1.0564 & 1 & 0 & 0 \\ -0.3169 & 0 & 1 & 0 \\ -0.1556 & 0 & 0 & 0 \\ 0.1960 & 0 & 0 & 0 \end{bmatrix} \underline{x[k]} + \begin{bmatrix} 1 \\ -0.2 \\ -0.93 \\ 0.27 \end{bmatrix} u[k]$$

$$\underline{y[k]} = [1\ 0\ 0\ 0\ ]\underline{x[k]} + 0\ u[k]$$

(N. B. T_s = 0.1 secs.)

Demonstration system #9 is a single-input single-output system with two complex roots. One of the real roots is unstable. The discrete-time system has a zero on the negative real axis of the Z-plane. Only the discrete-time state-variable model (SYS9D) is available.

Demonstration System #F4

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -0.0068 & 0.0015 & -0.6594 & -0.3200 & 1 \\ 00011 & -0.4940 & 14.4840 & -0.0145 & 0 \\ 0.3410 & -1.9800 & -0.4880 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.1 \end{bmatrix} \underline{x}(t)$$

$$+ \begin{bmatrix} 0.0321 & 0 \\ -0.8060 & 0 \\ -15.99 & 0 \\ 0 & 0 \\ 0 & 0.1 \end{bmatrix} u(t)$$

$$\underline{y}(t) = \begin{bmatrix} 1.033 & 0 & 0 & 0 & 0 \\ 0 & 0.0689 & 0 & 0 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u}(t)$$

Discrete State Variable Model

$$\underline{x}(k+1) = \begin{bmatrix} 0.9994 & 0.0017 & -0.0325 & -0.0160 & 0.0499 \\ 0.0061 & 0.9408 & 0.6981 & -0.0007 & 0.0001 \\ 0.0166 & -0.0954 & 0.9408 & -0.0001 & 0.0004 \\ 0.0004 & -0.0024 & 0.0488 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0.9950 \end{bmatrix} \underline{x}(k)$$

$$+ \begin{bmatrix} 0.0147 & 0.0001 \\ -0.3175 & 0 \\ -0.7787 & 0 \\ -0.0197 & 0 \\ 0 & 0.0050 \end{bmatrix} \underline{u}(k)$$

$$\underline{y}(k) = \begin{bmatrix} 1.033 & 0 & 0 & 0 & 0 \\ 0 & 0.0689 & 0 & 0 & 0 \end{bmatrix} \underline{x}(k) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u}(k)$$

(N. B.: $T_s = 0.05$ secs.)

Demonstration system #F4 is a multi-input multi-output system with five roots. It is a linearized model of the pitch dynamics of the F4 aircraft. The model was obtained from the MATRIXx User's Manual. A continuous state-variable model (SYSF4S), and a discrete-time state-variable model (SYSF4D) are available.

Demonstration System #F15

Continuous State Variable Model

$$\dot{\underline{x}}(t) = \begin{bmatrix} -0.002 & -0.307 & 0 & -0.322 & 0 & 0 & 0 & -0.122 & 0 & 0.982 \\ -0.001 & -0.342 & 1 & 0 & 0 & 0 & 0 & -0.54 & 0 & 0.002 \\ 0.024 & -1.548 & -0.989 & 0 & 0 & 0 & 0 & -3.8 & 0 & 0.001 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.364 & -23.994 & 0.170 & 0 & -0.333 & 0 & 0 & -58.9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 \\ 0.016 & -3.366 & -0.872 & 0 & 0 & 0 & 0 & -3.668 & 0 & 0 \\ 0.006 & -0.369 & -0.236 & 0 & 0.349 & 0.349 & 0.349 & -20.9060 & 0 & 0 \\ 0.101 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.1 \end{bmatrix} \underline{x}(t)$$

$$+ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 11.49 & 0 \\ 0 & 0 \\ -1.562 & 0 \\ 0 & 0 \\ 0 & 0.1 \end{bmatrix} u(t)$$

$$\underline{y}(t) = \begin{bmatrix} -0.004 & -1.61 & 0 & 0 & 0 & 0 & 0 & 0.255 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u}(t)$$

(Continued)

Discrete State Variable Model

$$\underline{x(k+1)} =$$

$$\begin{bmatrix} 0.999 & -0.015 & -0.001 & -0.016 & 0 & 0 & 0 & -0.004 & 0 & 0.049 \\ 0 & 0.981 & 0.048 & 0 & 0 & 0 & 0 & -0.005 & 0 & 0.001 \\ 0.001 & -0.073 & 0.951 & 0 & -0.001 & -0.001 & -0.001 & -0.113 & 0 & 0 \\ 0 & -.002 & 0.049 & 1 & 0 & 0 & 0 & -0.003 & 0 & 0 \\ 0.018 & -1.151 & -0.008 & -0.001 & 09653 & -0.017 & -0.018 & -1.792 & 0 & 0.001 \\ 0 & 0 & 0 & 0 & 0 & 0.779 & 0 & 0 & 0 & 0 \\ 0.001 & -0.163 & -0.046 & 0 & -0.001 & -0.001 & 0.999 & -0.110 & 0 & 0 \\ 0.003 & -0.019 & -0.008 & 0 & 0.011 & 0.009 & 0.011 & 0.338 & 0 & 0 \\ 0.005 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.995 & 0.001 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.995 \end{bmatrix} \underline{x(k)}$$

$$+ \begin{bmatrix} 0.001 & 0.001 \\ 0.001 & 0 \\ 0.005 & 0 \\ 0.001 & 0 \\ 0.791 & 0 \\ 0.508 & 0 \\ 0.005 & 0 \\ -0.045 & 0 \\ 0 & 0 \\ 0 & 0.005 \end{bmatrix} u(k)$$

$$\underline{y(k)} = \begin{bmatrix} -0.004 & -1.61 & 0 & 0 & 0 & 0 & 0 & 0.255 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \underline{x(t)} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \underline{u(k)}$$

(N. B.: T_s = 0.05 secs.)

Demonstration system #F15 is a multi-input multi-output system with five roots. It is a linearized model of the pitch dynamics of the F15 aircraft. The model was obtained from an informal report by Integrate Systems Inc. as a part of their trajectory autopilot study. A state-variable model (SYSF15S), and a discrete-time state-variable model (SYSF15D) are available.

Demonstration System #X29

Continuous State Variable Model

$$\dot{x}(t) = \begin{bmatrix} -1.42E\text{-}2 & -8.01 & -0.2078 & -32.08 \\ -1.941E\text{-}4 & -0.5246 & 0.9953 & 0 \\ 2.985E\text{-}4 & 7.59 & -0.2282 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x(t)$$

$$+ \begin{bmatrix} -0.118 & -0.1166 & 8.063E\text{-}3 & 14.09 \\ -1.558E\text{-}4 & -2.587E\text{-}3 & -4.063E\text{-}4 & -1.731E\text{-}3 \\ 4.68E\text{-}2 & -2.98E\text{-}2 & -1.98E\text{-}2 & -9.99E\text{-}2 \\ 0 & 0 & 0 & 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 57.3 & 0 & 0 \\ 0 & 0 & 57.3 & 0 \\ 0 & 0 & 0 & 57.3 \\ 3.64E\text{-}3 & 9.981 & 8.07E\text{-}2 & -2.03E\text{-}4 \\ 3.62E\text{-}3 & 9.549 & 9.36E\text{-}2 & -2.03E\text{-}4 \end{bmatrix} x(t)$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 4.72E\text{-}3 & 4.71E\text{-}2 & 6.83E\text{-}3 & -3.36E\text{-}3 \\ 2.04E\text{-}3 & 4.88E\text{-}2 & 7.96E\text{-}3 & 3.34E\text{-}3 \end{bmatrix} u(t)$$

Demonstration system #X29 is a multi-input multi-output system providing a linearized models of the pitch dynamics of the X29 aircraft. A continuous state-variable model (SYSX29A), and a discrete-time state-variable model (SYSX29D) are available. The continuous model is a fourth-order model of the bare airframe with two complex and two real roots. One of the real roots is unstable. The discrete-time model is a 45th-order model of the complete aircraft. The discrete-model is available on disk, but is not presented in this paper.

MODRED USER'S GUIDE

Appendix D

Bibliography

Bibliography

1. R. C. Dorf; "Modern Control Systems", third edition, Addison-Wesley, 1980.

2. G. F. Franklin and J. D. Powell; "Digital Control of Dynamic Systems", Addison-Wesley, 1980.

3. S. C. Shah, R. A. Walker, and C. Z. Gregory jr.; "MATRIX_ User's Guide", Integrated Systems, Inc., Palo Alto, Ca., 1982.

4. R. A. Walker and N. Gupta; "Flight Test Trajectory Control Analysis", Integrated Systems, Inc., Palo Alto, Ca., 1982.

5. Per-Olof Gutman, Carl Frederick Mannerfelt, and Per Molander; "Contributions to the Model Reduction Problem", IEEE Transactions on Automatic Control, Vol. AC-27, No. 2, Apr. '82, pp. 454-455.

6. B. C. Moore; "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction.", IEEE Transactions on Automatic Control, Vol. AC-26, No. 1 Feb. '81, pp. 382-387.

NASA National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
CR-179434		

4. Title and Subtitle	5. Report Date
Model Reduction Methods for Control Design	August 1988
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
K.R. Dunipace	H-1499
	10. Work Unit No.
	RTOP 505-66-11

9. Performing Organization Name and Address	11. Contract or Grant No.
Engineering Division Indiana University—Purdue University at Indianapolis West Lafayette, Indiana 47907	NCC2-289
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report—Final
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

NASA Technical Monitor: Eugene L. Duke, NASA Ames Research Center, Dryden Flight Research Facility, Edwards, California 93523-5000.

16. Abstract

This report develops several different model reduction methods and provides detailed implementation information for those methods. The report contains command files to implement the model reduction methods in a proprietary control law analysis and design package. A comparison and discussion of the various reduction techniques is included.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Flight control Linear models Model reduction	Unclassified—Unlimited Subject category 63

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	140	A07

*For sale by the National Technical Information Service, Springfield, Virginia 22161.

End of Document